

1|85 DAS MAGAZIN FÜR COMPUTER-FANS

Eine Markt & Technik Publikation

Commodore-Heimcomputer

Gestern – heute – morgen

Die VC 20-Nachfolger

C 16/C 116 im Test

Viel Computer für wenig Geld

C 64 als Sporttrainer

Die 7 besten

Assembler im Vergleichstest

2000 Mark für das Listing des Monats

Mit Pinsel und Palette

Ein Superprogramm zum Malen und Zeichnen

Neuer Kurs:

Effektives Programmieren

Alles über Strings

1985 ?

1984 Plus/4

1984 C 16/C 116

1983 SX 64

1982 C 64

1981 VC 20

Test: GBasic ★ VC 20 als C 64 ★ Die neue Art zu Lernen: Softlearning ★ Forth in der Praxis ★ Neues über Hypra-Load ★ Tips und Tricks für den VC 20 und C 64

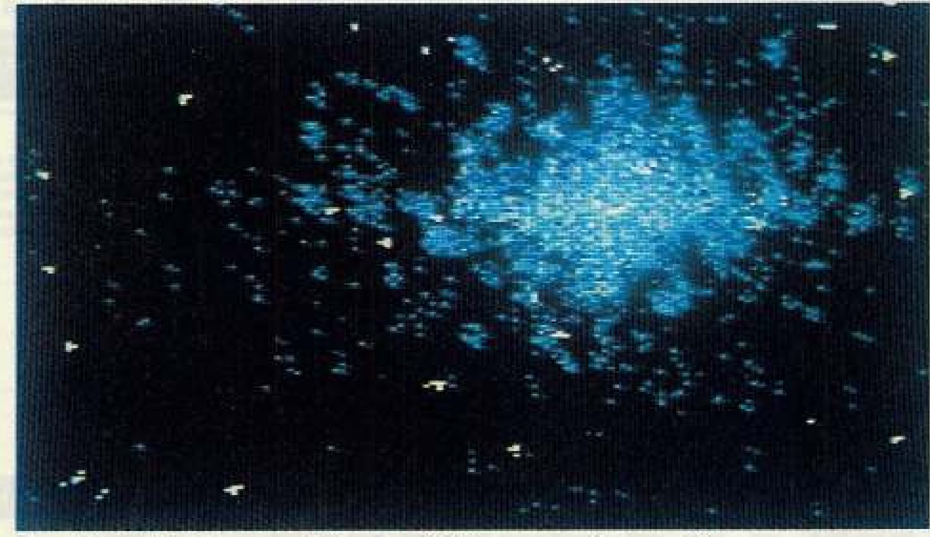
INHALT



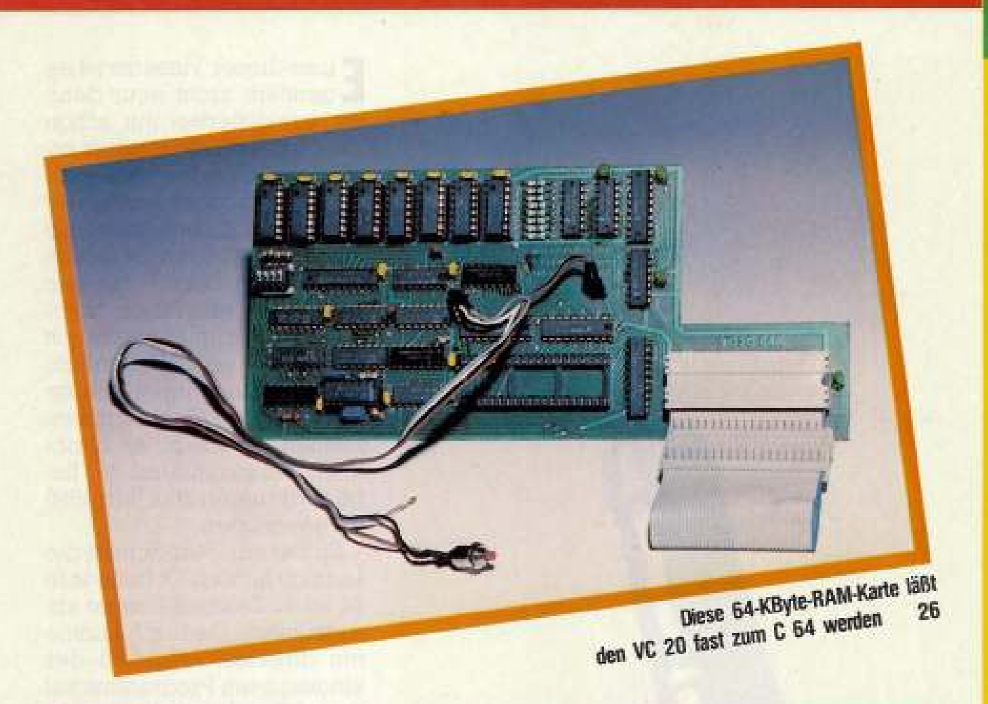
Der C 16 mit neuen Möglichkeiten soll den VC 20 ersetzen 16



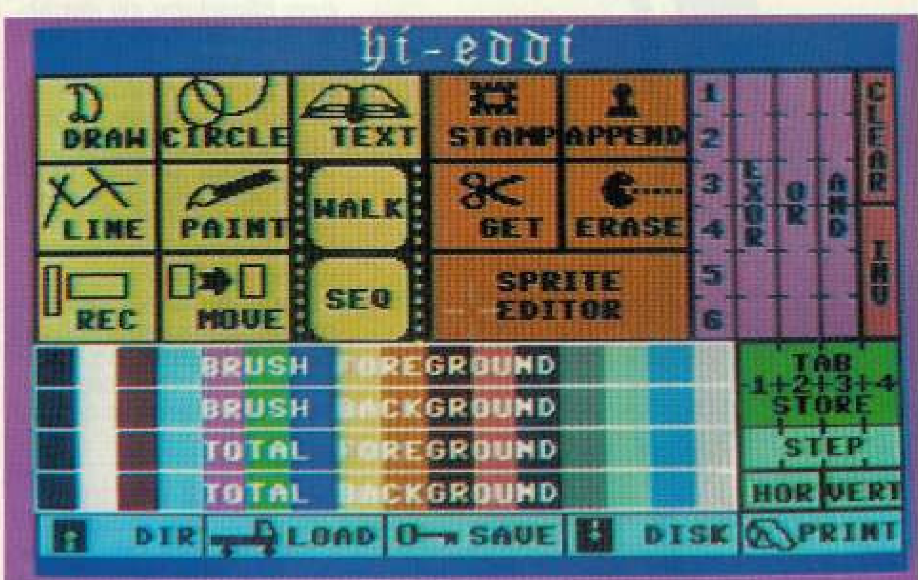
Ist GBasic die ideale Spracherweiterung für den C 64? 28



Die weiche Welle des Lernens — Softlearning mit »Entspannungsgrafiken« 40



Diese 64-KByte-RAM-Karte läßt den VC 20 fast zum C 64 werden 26



HI-EDDI als Listing des Monats: Ein Zeichen- und Malprogramm das sich nicht hinter den »professionellen« zu verstecken braucht 50



Provic 64 — Vier Bildschirmfenster mit bis zu 32 Sprites 76

Programme zum Abtippen

Anwendung

Grafik

Tips & Tricks

Kurse

Rubriken



Lohnen sich Clubs?

Commodore 64-Clubs sind in den letzten zwölf Monaten wie Pilze aus dem Boden geschossen. Ihre Ziele sind weitgehend ähnlich: Erfahrungs-, Literatur- und Programmaustausch — hin und wieder kommen dazu Fortbildung, gemeinsamer Einkauf oder Bildung von Arbeitsgruppen zum Schreiben bestimmter Programme oder für gemeinsame Hardware-Bastelei.

Lohnen sich solche Aktivitäten eigentlich für die Mitglieder? Kann man durch Sammelbestellungen — und wenn ja, bei welchen Produkten — wirklich wesentlich günstigere Preise herausholen, als durch Vergleich der Preise mehrerer Einzelhandelsgeschäfte und Versandhändler? Lohnt sich der Versuch, mit einer bestimmten Firma einen Rabatt für Clubmitglieder auszuhandeln oder achtet man lieber darauf, wo es gerade das eine oder andere Sonderangebot gibt? Wo sind Beispiele für besondere Erfolge, die in einer Arbeitsgruppe erzielt wurden? Welche ausländischen Benutzergruppen sind wegen ihrer Programmbibliothek oder aus anderen Gründen besonders interessant? Solche Fragen drängen sich dem Benutzern auf.

Ich würde mich freuen, wenn Sie uns Ihre Erfahrungen schreiben würden. Wir werden in einer der nächsten Ausgaben in einem Bericht darauf eingehen.

Michael Pauly, Chefredakteur



Das Jahr 1985 steht vor der Tür, und wie das so üblich ist, nimmt man sich für ein neues Jahr immer etwas vor. Sei es, sich das Rauchen abzugewöhnen, oder freiwillig auf der Autobahn nicht schneller als 100 km/h zu fahren oder sonstige, unmögliche Dinge. Auch wir haben natürlich eine Menge guter Vorsätze.

Einer dieser Vorsätze ist eigentlich nicht neu, denn wir praktizierten ihn schon 1984, nämlich ein rundum zufriedenstellendes Magazin für Commodore-Anwender zu machen. Zu beurteilen, ob uns das gelungen ist, bleibt letztendlich eben diesem Anwender, also Ihnen, überlassen. Wir als Redaktion meinen — nein. Sicherlich, es sind schon einige Schritte in die richtige Richtung zu erkennen, trotzdem, es bleibt noch viel zu tun. Und das haben wir uns für das Jahr 1985 vorgenommen.

So war ein '84er-Schritt, die Listings für den Commodore 64 im 40 Zeichen-Format abzudrucken. Dadurch konnte ein direkter Vergleich des eingetippten Programms auf dem Bildschirm mit der abgedruckten Version vollzogen werden. Bei Programmen mit vielen DATA-Zeilen bauten wir relativ sichere Prüfsummen ein. Doch all das erwies sich als noch nicht ausreichend, wie uns die vielen Anrufer bestätigten, die immer noch Schwierigkeiten beim Eintippen der Programme hatten. Probleme gab es hauptsächlich bei der Identifizierung der Steuer- und Grafikzeichen, bei DATA-Zeilen und den VC 20-Listings.

Sichere Listings

Deshalb sind ab dieser Ausgabe alle Listings ohne Steuer- und Grafikzeichen abgedruckt sowie mit einer zeilenweisen Prüfsumme versehen. Wie das im einzelnen aussieht, und vor allen Dingen wie in Zukunft unsere Listings abzutippen sind, lesen Sie bitte in dem Artikel »Checksummer — nie mehr Probleme beim Abtippen«. Eins ist auf jeden Fall gewährleistet. Sie wissen nun nach jeder Zeile, ob sie richtig eingegeben wurde. Damit gehört das unvermeidlich scheinende SYNTAX ERROR IN nach dem ersten Start der Vergangenheit an. Die Lupe für die Steuerzeichen kann man sich fürderhin auch sparen, die Herzchen und Strichlein gibt's nicht mehr. Also nicht vergessen, zuerst den Check-

summer eintippen (entweder für C 64 oder VC 20) und dann erst all die anderen interessanten Listings in dieser Ausgabe.

In vielen Zuschriften ist immer wieder zu lesen: »Macht weiter so!«. Genau das haben wir nicht vor. Denn würden wir auf unserem heutigen Stand verharren, der zugegebenermaßen nicht schlecht ist, bliebe kein Platz für weitere Verbesserungen. Und das ist es, was wir noch anstreben.

Lange lebe der C 64

Denn wen stört es nicht, die ewigen DATA-Wüsten abzutippen. Wir arbeiten zur Zeit an einem System, das das Eintippen von Maschinenprogrammen erheblich vereinfacht. Nur noch Zweidrittel des Volumens und etwa die Hälfte der Zeit werden in der neuen Form benötigt. Lassen Sie sich überraschen.

Der Übersichtlichkeit soll auch die neue Gestaltung der einzelnen Rubriken dienen. Die kompaktere Form der Kurse ermöglicht es uns, mehr Information auf gleichem Raum unterzubringen. Wie auch im vergangenen Jahr bitten wir unsere Leser um rege Mitarbeit. Nur wenn wir wissen, wo der Schuh drückt, können wir entsprechend darauf reagieren.

Speziell unsere VC 20-Leser sind besorgt, daß wir den VC 20 bald vergessen werden. Die VC 20-Anhänger können beruhigt sein, solange ihr Computer in großen Stückzahlen auf dem Markt ist, bleibt er in der Berichterstattung. Die Art der Artikel wird sich aber nach dem abzusehenden Einsatzgebiet als preiswerter Meß-, Steuer- und Regelcomputer richten.

Der Nachfolger für den VC 20, der C 16/116 ist mittlerweile in den Läden. Dies ist der ideale Einsteigercomputer mit einem sehr mächtigen Basic, der sich ebenfalls zu einem Renner entwickeln könnte. Wieder in Relation zu den verkauften Stückzahlen wird er von uns in Zukunft mit entsprechenden Beiträgen bedacht werden.

So wie der C 16 als direkter Nachfolger des VC 20 angesehen werden kann, sind viele auch der Auffassung, der Plus/4 wäre der Nachfolger des C 64. Diese Meinung ist nicht ganz richtig. Der Plus/4 soll den C 64 weder ersetzen noch erweitern. Er richtet sich an eine andere Zielgruppe, wie die Leistungsmerkmale und der Preis erkennen lassen. Man sollte also den Plus/4 völlig unabhängig vom C 64 sehen. Zum C 64 mehren sich mittlerweile die Anfragen, ob dieser Computer schon veraltet sei und sich ein Kauf noch lohnen würde. Ähnliche Fragen wurden, den VC 20 betreffend, bereits vor zwei Jahren an Computer persönlich gerichtet. Die Antwort können Sie aus dem langen Zeitraum schließen, den der VC 20 von da an noch »überlebt« hat. Zudem besitzt der C 64 wesentlich bessere Voraussetzungen, im harten Konkurrenzkampf zu bestehen. Erst wenn es Heimcomputer mit 16 Bit-Prozessoren unter 1000 Mark gibt, lohnt sich die Überlegung, den C 64 nicht mehr zu kaufen. Bis dahin dürfte aber auch oder gerade Commodore reagiert haben. (aa)

Die ersten 500 000 Commodore 64 verkauft

Am 15. November gab es im Karstadt-Computer-Center in Dortmund ein Jubiläum zu feiern. Der fünfhunderttausendste in Deutschland verkaufte C 64 wurde dort seinem Besitzer, Dr. Wolfgang Kuhn aus Recklinghausen, übergeben. Der Jubiläumskunde erhielt dabei nicht nur den Kaufpreis des Computers erstattet, sondern auch noch Disketten-Laufwerk, Farbmonitor und einen großen Sack voll Software geschenkt. Darüber hinaus spendete Karstadt zehn weitere C 64 für den Informatik-Unterricht an zwei Dortmunder Schulen. Mit 500 000 verkauften C 64 alleine in der Bundesrepublik und mehr als drei Millionen weltweit verkauften Geräten, ist der C 64 der erfolgreichste Computer aller Zeiten, gefolgt vom VC 20 mit insgesamt 1,5 Millionen verkauften Exemplaren. Dr. Kuhn, selbst Lehrer an einer berufsbildenden Schule, begründetete seine Entscheidung für den C 64 mit dem Hinweis, daß er endlich einmal den Wissensvorsprung seiner Schüler auf diesem Gebiet aufarbeiten wolle. Immer häufiger habe er von seinen Schülern Sätze zu hören bekommen wie »Aber Herr Kuhn, warum machen Sie das so umständlich? Mit meinem C 64 löse ich das viel schneller«.

Eine Aussage, die vermutlich symptomatisch für die derzeitige Situation im Schulwesen sein dürfte. (ev)



Der 500 000ste Commodore 64-Besitzer

Commodore an Schulen

Nachdem der C 64 an vielen Gymnasien bereits der Standardcomputer für den Informatik-Unterricht ist, plant Commodore nach Aussage von Alwin Stumpf, Geschäftsführer Vertrieb Deutschland, nun die Einführung des C 64 speziell an Haupt-, Real- und berufsbildenden Schulen. Dabei ist allerdings nicht daran gedacht, an diesen Schulen einen Informatik-Unterricht einzuführen (»Wir brauchen kein Volk von Informatikern«), sondern es geht darum, auf die praktische Arbeit mit dem Computer vorzubereiten. Dazu gehört in erster Linie auch, den zukünftigen Lehrlingen und Praktikanten die Scheu vor dem Computer zu nehmen.

In Zusammenarbeit mit dem Westermann-Schulbuch-Verlag, der entsprechende Lernsoftware entwickelt, werden derzeit die Grundlagen für die breite Einführung des C 64 an Haupt- und Realschulen geschaffen. Der Computer soll ähnlich Diaprojektoren, Film und Video, einfach ein weiteres technisches Medium zur Vermittlung von Unterrichtsinhalten werden. (ev)

64-Erweiterungen

XL80 heißt eine 80-Zeichen-Karte für den C 64, die General Automation vertreibt. Im Preis von zirca 400 Mark ist außerdem ein Textverarbeitungs- und ein Tabellenkalkulationsprogramm enthalten. Außerdem wurde ein Wafertapelaufwerk für den C 64 angekündigt, das bei Lade- und Speichervorgängen 10- bis 20mal schneller als die Datasette und immerhin noch dreimal so schnell wie das 1541-Diskettenlaufwerk sei. Zwei dieser Wafertape-Laufwerke könnten hintereinander geschaltet werden. Mitgeliefert wird zum Preis von zirka 200 Mark pro Laufwerk auch die notwendige Software einschließlich Kopierprogrammen. Auf ein Endlosband (Wafertape, Preis: zirka 9 bis 10 Mark) passen 128 KByte. (sc)

Info: General Automation, Hagenauer Str. 42, 6200 Wiesbaden, Tel. (06121) 23093

Ein wenig mehr fürs 64'er

Zum Jahresbeginn müssen wir den Preis fürs 64'er den gestiegenen Papier- und Herstellungskosten anpassen: Ab Ausgabe 1/85 beträgt der Einzelpreis 6,50 Mark; die Kosten für das Abonnement (12 Ausgaben) betragen ab 1. Januar 78 Mark. Vorteil für Abonnenten: Sie bekommen die Zeitschrift regelmäßig ins Haus geschickt und sind damit sicher, daß Sie keine Ausgabe verpassen.

COMMODORE

Wer hätte das vor zwei, drei Jahren zu denken gewagt? Ein Hersteller hält 50 Prozent des Marktes bei den Heimcomputern. Commodore hat es geschafft. Angefangen hat das mit Schreibmaschinen. Was jedoch noch viel interessanter ist, wie wird es weitergehen?

Er war eigentlich schon längst überfällig, der Bericht über die Entstehungsgeschichte von Commodore. Zum Jahreswechsel sollen deshalb der Werdegang und die Zukunftsaussichten dieses faszinierenden Unternehmens beschrieben werden.

Vor rund 26 Jahren wurde ein kleiner Verkaufs- und Reparaturladen für mechanische Schreibmaschinen in Toronto, Kanada, in Commodore International Limited umbenannt. Damit war der Name für den erfolgreichsten Personal- und Heim-Computer-Hersteller geboren. Inhaber dieser kleinen Firma war Jack Tramiel. Tramiel ist polnischer Abstammung, überlebte das Konzentrationslager und wanderte nach Nordamerika aus. Im Jahre 1958 hat er dann den kleinen Schreibmaschinenladen aufgemacht. Diesem Tramiel dürfte auch der Erfolg der ersten 25 Jahre zu verdanken sein. Er wird als ein Mann beschrieben der die Fähigkeit hat, im Bereich der Elektronik für Heim und Beruf die zukünftigen Bedürfnisse zu erkennen, und was noch viel wichtiger ist, rechtzeitig darauf zu reagieren. Das ist der Punkt, der Commodore auch heute noch auszeichnet.

Während dieser ersten Jahre entwickelte sich Commodore von einem Verkaufs- und Reparaturladen durch den Zukauf einer Berliner Firma, zum Hersteller für eben diese Schreibmaschinen. Anfang der 60er Jahre bot die Firma eine große Palette an Büroausrüstungen an, zudem übernahm sie den Vertrieb von Büromöbeln.

1965 kaufte Commodore die Möbelfirma, deren Distributor sie bisher war, auf und zog in deren Bürogebäude ein. Commodore stellt übrigens immer noch Büromöbel her, hauptsächlich Schränke und Tische sowie die Gehäuse für die CBM-Serie. Im selben Jahr lernte Tramiel den kanadischen Rechtsanwalt und Bankier Irvin Gould kennen, den jetzigen Präsidenten von Commodore. Diese beiden machten Commodore zu der Firma, die sie heute ist. Eines der ersten Aktionen dieses Führungsteams war es, die Firma für mechanische Addiermaschinen zu verkaufen, mit der Absicht sich einen japanischen Hersteller zu angeln, für den sie den Vertrieb übernehmen könnten. Während seines Aufenthalts in Japan bekam Tramiel dabei zum erstenmal eine elektronische Addiermaschine zu Gesicht. Er erkannte sofort, daß dieses Ding das Aus für die mechanischen Addiermaschinen bedeuten würde. Mit der auch heute noch gültigen Commodore-Philosophie »Wenn Du nicht Dein eigener Konkurrent bist, werden es andere für Dich sein« gab Tramiel die Suche nach dem mechanischen Addiermaschinenhersteller auf, fand dafür aber eine Firma die bereit war, unter dem Namen Commodore elektronische Rechner herzustellen.

1969 lief die Produktion in eigenen Werkstätten an. Man benötigte allerdings die ICs von Texas Instruments dafür. Damit gelang es Commodore aber, auch den ersten »richtigen« Taschenrechner — C108 genannt — mit sage und schreibe den gesamten vier Grundrechenarten auf den Markt zu bringen. Interessanterweise kostete dieser Taschenrecher damals genausoviel wie der C 64 heute. Bis 1974 weitete Commodore die Produktpalette der Taschenrechner kontinuierlich aus. Es kamen spezielle technisch-wissenschaftliche Rechner, »richtige«, programmierbare Computer und Speichereinheiten hinzu. Zu dieser Zeit war Commodore extrem abhängig von Zulieferungen der IC- und Anzeigenhersteller.

Schritte zur Unabhängigkeit

Dann kam auch noch hinzu, daß 1975 Texas Instruments mit der Produktion von eigenen Taschenrechnern begann und damit als direkter Konkurrent des früheren Kunden auftrat. Zur selben Zeit setzte der Preisverfall bei den Chips ein und Commodore sah sich einem großen Lagerbestand an (teuer eingekauften) ICs und Rechnern gegenüber, während die Marktpreise fielen und die Konkurrenz wuchs. Diese Situation führte bei Tramiel zu der Entscheidung, nie wieder von Dritten abhängig zu sein. 1976 erwarb Commodore deshalb MOS Technology, einen ehemaligen Halbleiterzulieferanten. 18 Monate später folgte der Kauf des Chip-Herstellers Frontier in Los Angeles, dessen Produktspektrum eine ideale Ergänzung zu den Erzeugnissen von MOS darstellte. Zur Komplettierung diente dann noch der Erwerb von Micro Display Systems, einem Hersteller von Flüssigkristall-Anzeigen in Dallas. Durch diese Zukäufe sammelte sich bei Commodore mehr Know-How und Produktionskapazität in den wichtigsten Schlüsseltechnologien an, als es bei wesentlich größeren Konkurrenten der Fall war. Dieses Alles-in-einem-Haus-Konzept ermöglichte es Commodore, speziell für die eigenen Bedürfnisse zu entwickeln und zu produzieren. Dadurch hatte und hat Commodore einen nicht unerheblichen Zeitvorsprung bei der Entwicklung neuer Produkte und der Kosteneinsparung durch rationelle Produktion, die sich natür-



Der neue C 16 als Nachfolger für den VC 20

GESTERN Heute MORGEN

lich im Preis/Leistungsverhältnis zu Gunsten des Verbrauchers niederschlugen.

1977 kam dann der große Durchbruch, der erste richtige Personal Computer mit dem Namen PET wurde vorgestellt. Der PET (Personal Electronic Transactor) besaß bereits den von MOS konstruierten 6502-Prozessor, den auch einige Mitbewerber in ihre Systeme einbauten. Dieser PET kann als der Großvater des heutigen Heim- und Personal-Computermarktes angesehen werden. Der PET wurde weltweit vertrieben und erreichte gerade in Europa durch Schulen, Universitäten und den heimischen Wohnzimmern eine große Akzeptanz. Es folgten die Serien CBM 4000 und CBM 8000, die sich durch einen größeren Speicherplatz auszeichneten. Sie waren jedoch mehr auf den Profi-Markt ausgelegt, und zielten nicht so sehr auf den Heimbereich.



Hat die neue Konzeption des Plus/4 eine Zukunft?

Dieser wurde 1981 von dem »Volkscomputer« VC 20 abgedeckt. Über den Erfolg des weltweit mehr als zweimillionenmal verkauften VC 20 braucht eigentlich nichts gesagt zu werden.

Mit dem Erfolg des VC 20 gab sich Commodore aber noch nicht zufrieden. 1982 stellte man den C 64 vor. Ein System, das erstmals mit der magischen Zahl 64-KByte RAM-Speicher zu einem vertretbaren Preis aufwarten konnte. Auch über dieses System zu schreiben ist müßig, die meisten Leser wissen bestens bescheid über diesen Computer.

Anfang 1984 gab es dann ein großes Zerwürfnis bei Commodore. Jack Tramiel verließ »seine« Firma. Die Gründe sind vielfältiger Natur. Die Fähigkeiten von Tramiel sind unbestritten. Er war es in der Vergangenheit, der Commodore zu dieser Weltstellung verhalf. Auf der anderen Seite, hatte er sein Unternehmen mehr oder weniger als Diktator geleitet. Entscheidungen wurden grundsätzlich in der Firmenspitze getroffen, also von Tramiel. Als er aber versuchte seine Söhne in die höheren Posten bei Commodore einzuschleusen regte sich Widerstand, nicht zuletzt von Irvin Gould. So kam es zu dem großen Bruch, Tramiel verließ Commodore. Er kaufte sich die marode Abteilung Atari von Warner Corporation und versucht seitdem diesem Computer- und Spielemodulhersteller wieder auf die Beine zu helfen. Man munkelt aber bereits von erheblichen Finanzierungsschwierigkeiten. Dennoch herrscht in den USA die Meinung vor, man sollte Tramiel nicht unterschätzen und könne gespannt sein, was in Zukunft von Atari zu erwarten sei.

Zu erwarten ist von Atari sicherlich nicht mehr der Amiga-Computer. Atari stand mit der Firma Amiga in Verhandlungen, deren Neuentwicklung, einen Macintosh-ähnlichen Computer zu vertreiben. Dieser Absicht ist Commodore mit dem kompletten Aufkauf der Firma Amiga zuvorgekommen. Es wird erwartet, daß dieser »Amiga« bereits im Herbst 1985 in Deutschland erhältlich sein soll. Die angekündigten Leistungsmerkmale lassen einiges erhoffen: höhere Bildschirmauflösung als der Macintosh und das bei farbiger Darstellung, schnellerem Prozessor, größerer Speicherkapazität und einem Preis der nur die Hälfte des Mac betragen soll. Dem hat Atari nichts gleichwertiges entgegenzusetzen. Im Gegenteil, der Ausverkauf des 800 XL mit Dumping-Preisen beweist, daß Atari momentan in erheblichen Schwierigkeiten steckt.

Zu den neuen Computern C16/C116 und Plus/4 von Commodore hat sich der Commodore-Guru Jim Butterfield äußerst positiv ausgelassen. In einem Interview anläßlich der CFA (Commodore Fachausstellung in Frankfurt) vertrat er die Auffassung, daß diese Computer durchaus ihre Chance auf dem Markt haben werden. Der C 16/C 116 ist der direkte Nachfolger für den Veteran VC 20. Warum sich die Anschaffung dieses Einsteiger-Computers lohnt, können Sie im Testbericht in dieser Ausgabe nachlesen. Zum Plus/4 werden wir einen ausführlichen Testbericht in der nächsten Ausgabe bringen. Eine weit verbreitete Meinung ist, daß der Plus/4 der direkte Nachfolger des C 64 sei, wie dies der C 16 für den VC 20 ist. Man ist, als Anhänger dieser Meinung, dann natürlich enttäuscht. Der Plus/4 hat ja im Vergleich zum C 64 wesentlich weniger in Bezug auf Grafik (keine Sprites) und Ton (nur eine Stimme) zu bieten. Die eingebauten Programme sind zwar bei Commodore ein Novum, aber er ist nicht einmal kompatibel zum C 64.

Das soll er wohl auch nicht sein, denn er ist nicht der Nachfolger des C 64. Der Plus/4 soll den Anwenderbereich nach oben hin ergänzen, und den C 64 nicht ersetzen. Er ist also nicht für den typischen Heimbereich konzipiert, sondern mehr für die semiprofessionelle Anwendung in Betrieben kleiner und mittlerer Größe. Wahrscheinlich hat Commodore bei diesen Anwendern noch eine Marktlücke entdeckt, und gedenkt diese mit dem Plus/4 abzudecken. Ob dem so ist, wird die Zukunft zeigen.

Während sich für den C 16/C 116 eine glänzende Zukunft voraussagen läßt, bleibt beim Plus/4 eine gewisse Skepsis angebracht. Vor allen Dingen die Nicht-Kompatibilität mit dem C 64 steht dem wohl im Wege.

Doch wie ich Commodore kenne, wird mit einen einwandfrei zu bestimmenden Nachfolger für den C 64 im Jahre 85 oder 86 zu rechnen sein. Diese Ausssage gilt natürlich nur für den amerikanischen Markt. Es kann

durchaus noch dieses »neue« Jahr in Deutschland verstreichen, bis der »Nachfolger« auch bei uns in Stückzahlen erhältlich ist. Ob er dann kompatibel zum C 64 ist, welche Leistungsmerkmale ihn auszeichnen, wieviel er kosten wird, bleibt abzuwarten. Wird er 128 oder 256 KByte RAM zur Verfügung stellen, welcher Prozessor ist eingebaut, ist er hundertprozentig kompatibel oder gar CP/M-beziehungsweise MS-DOS-fähig? Das bringt wohl alles das Jahr 1985 zutage.

Was ist also 1985, im Jahr nach der berühmten orwellschen Wendemarke, zu erwarten? Auf jeden Fall der »Amiga«, ein Computer, der dem Macintosh oder gar der Lisa das Leben schwer machen wird. Dann der C 16/C 116 mit glänzenden Zukunftschancen als Einsteigercomputer. Der Plus/4, von dem man noch nicht genau weiß, was man von ihm halten soll. Und schließlich der sehnlichst erwartete Nachfolger des C 64. Kommt er oder kommt er nicht? Das ist hier die Frage. (aa)

Floppy-Laufwerk: Zweiter Anlauf

Das in der Ausgabe 9/84 vorgestellte Floppy-Laufwerk YL-55S1-CM von expuls durfte aus lizenztechnischen Gründen nicht auf den Markt eingeführt werden. Die Gründe sind bekannt. Expuls hat daher eine Alternative für das 1541-Alternativ-Laufwerk entwickelt, das Floppy-Laufwerk EPH 1001. Die Laufwerke stammen von NEC, der intelligente Controller ist ein deutsches Erzeugnis und auch die Endmontage und Qualitätskontrolle findet in diesem unserem Lande statt. Durch das Doppelkopf-Laufwerk ergibt sich eine wesentliche höhere Speicherkapazität von 408 KByte (formatiert, beidseitig beschrieben). Zwei LEDs zeigen an, auf welche Diskettenseite gerade zugegriffen wird. Der Controller besteht aus zwei Prozessoren, 6502 und µPD 765, letzterer ist direkt ansprechbar. Eine Diskettenseite wird einschließlich Formatieren und VERIFY in zirka zwei Minuten kopiert. Programme die im 1541-Format abgespeichert wurden, lassen sich mittels eines mitgelieferten Umkopierprogramms auf das EPH 1001-Format überspielen.

Der Preis für den Endverbraucher liegt bei 1498 Mark. Ein ausführlicher Testbericht folgt. (aa)

Info: expuls, St.-Anton-Straße 31, 4150 Krefeld 1, Tel. 02151-801300

Kaufhof und Quelle steigen voll ins Computergeschäft ein

Der Kaufhof will in einer bundesweiten Aktion in 42 Filialen Computer-Studios einrichten. Ein Beweggrund ist das Medium Computer »publikumsnah« der breiten Bevölkerung vorzustellen. Ein anderer wohl, sich einen Teil vom Kuchen des Computer-Markts abzuschneiden. Die Angebotspalette reicht von dem kleinen Heimcomputer bis zum professionellen Personal-Computer. All diese Systeme kann man in den Computer-Studios testen. Das Fachpersonal soll gewillt sein, auch »dumme« Fragen zu beantworten. Um das Prinzip »Alles aus einer Hand« zu verwirklichen, gibt es neben der Hardware auch Software, Peripherie-Geräte und die passende Literatur. Der kostenlose Katalog »Computer-Studio« soll dem Interessierten die Entscheidung in Ruhe zu Hause treffen lassen.

Ohne Gedränge und Zeitnot konnte man sich bisher auch bei Quelle per Katalog und Versand seinen Computer aussuchen und kaufen. Dem Trend der Zeit folgend, hat nun auch Quelle Computer-Shops in den Vertriebs-Filialen eingerichtet. (aa)

256-KByte-Speichererweiterung für den C 64

Von Softline wurde eine Speichererweiterung für den C 64 angekündigt, deren 256 KByte auch als RAM-Floppy verwendbar sind. Serienmäßig ist die Erweiterung mit 64 KByte bestückt, läßt sich aber in 64-KByte-Schritten ohne Lötarbeiten bis auf die besagten 256 KByte ausbauen. Bei Verwendung als RAM-Floppy (32 KByte werden in einer Sekunde geladen oder gespeichert!) sorgt ein Batterie-Back up bei einem Stromausfall für die Datensicherheit (bis zwei Stunden). Die mitgelieferte Software emuliert eine 1541-Floppy-Station und übernimmt die Verwaltung des größeren Speichers.

Der größere Speicherplatz ist natürlich besonders interessant bei Tabellenkalkulations- und Textverarbeitungsprogrammen. So soll die Erweiterung kompatibel zu Calc Result, Multiplan, WordPro, Paper Clip, Logo, Hesmon und anderen professionellen Programmen sein. Geräte, die am seriellen Bus und am User-Port angeschlossen sind, werden weiterhin voll unterstützt. Ein Testbericht folgt in einer der nächsten Ausgaben.(aa)

Info: Softline R. Alverdes, Schwarzwaldstraße 8a, 7602 Oberkirch, Tel. 07802-3707

Commodore weiter im Aufwind

Ungehemmtes Wachstum kennzeichnete das vergangene Geschäftsjahr (1.7.83 bis 30.6.84) der Commodore Büromaschinen GmbH in Frankfurt. Diese positive Entwicklung setzte sich auch im ersten Quartal des neuen Jahres fort. So konnte der Gesamtumsatz um 104 Prozent auf 250,2 Millionen Mark gesteigert werden. Damit hat der Mikro-Marktführer seit seinem Bestehen das bisher beste Vierteljahres-Ergebnis erzielt. Die Resultate sind sogar deutlich besser als im legendären Weihnachtsquartal '83, als der Heimcomputer den endgültigen Durchbruch schaffte, erläuterte ein Firmensprecher. Die Zuwächse sind in erster Linie wohl auf die weiterhin ungebrochene Nachfrage nach den Commodore-Heimcomputern zurückzuführen. So wurde zum Beispiel am 15.11.1984 in Dortmund der 500000ste C 64 verkauft. Neben den Computern will Commodore auch das Angebot an Lernmedien, Begleit- und Arbeitsmaterial erheblich ausweiten. (aa)

Neuer 128-KByte-ROM-Speicher für den Commodore 64

Eine Platine für insgesamt 128 KByte ROM wurde dieser Tage von Frank Computertechnik, München vorgestellt. Die Platine ist für die Aufnahme von 8- bis 32-KByte-EPROMs vorgesehen. Es können so Speicherstufungen in 8-, 16-, 32-KByte-Schritten vorgenommen werden. Die Steuerung der einzelnen Speicherbereiche übernehmen zwei Register. Die Bedienung dieser Register kann in der Form eines Auswahlmenüs programmiert werden. Ein Directory mit dem Inhalt der Platine wird so angezeigt und das gewünschte Programm mit Knopfdruck gestartet. Da die Platine vollständig abgeschaltet ist, sollte einer dieser Wahlpunkte auch aus dem Sprung in das normale Basic bestehen. Die Platine beeinflußt dann das Laden von Basic- und Maschinenprogrammen in keiner Weise. Ein Verlust an Speicherplatz tritt in diesem Fall nicht auf. Die Beschreibung zur Platine ist sehr ausführlich und geht auch auf verschiedene Programmierschritte (Autostartkennung, Betriebssystem- und Basic-Initialisierung) ein. Die Platine ist sehr solide aufgebaut und wird mit einem stabilen Gehäuse geliefert. Der Preis soll bei zirka 100 Mark liegen. (gk)

Info: Frank Computertechnik, Metzstr. 8, 8000 München

Commodore 64 Super-Plus aus Holland

Commodore 64 Super Plus nennt sich eine erweiterte und verbesserte Version des C 64, die von der Rotterdamer Firma H&P-Computers angeboten wird. Sie beinhaltet schnellere Lade- und Speicher-Routinen sowohl für die Kassette (zehnmal schneller) als auch für die Floppy 1541. Durch ein geändertes Betriebssystem — es belegt keinen zusätzlichen Speicher — ist die Ladegeschwindigkeit fünffach höher und auch das Speichern von Programmen wurde auf ein Drittel der Zeit gekürzt. Dabei soll es weder bei Autostart noch bei mehrteiligen Programmen Schwierigkeiten geben. Insgesamt soll eine zirka 99prozentige Kompatibilität erreicht werden.

Ein Eingriff in die Floppy ist nicht notwendig, deswegen bleibt der serielle Port erhalten. Durch einen zusätzlichen Schalter auf dem C 64 kann ein Reset ausgelöst werden, der ein Basic-Programm nicht zerstört. Ebenfalls durch einen Schalter ist der normal- oder Super-Plus Modus schaltbar. Die Funktionstasten sind mit DOS-Befehlen belegt. Zu haben ist der C 64 Super-Plus für zusätzlich 250 Mark. Auch bereits vorhandene Computer können zum gleichen Preis nachgerüstet werden. (gk)

Bezugs- und Infoquellen für Deutschland, Österreich und Schweiz: v. Donkersloot, Verl. Parkweg 6, 6717 GN Ede, Tel. (08380) 32146

Wer hilft?

Wir sind eine kleine Gruppe von Computerfans und haben uns vorgenommen, einen Computerclub zu gründen. Leider wissen wir nicht, was man hierbei besonders beachten sollte (zum Beispiel, ob der Club angemeldet werden muß).

Harald Heckner
Spessartstr. 24
8400 Regensburg

Forth real hochauflösend?

Ich besitze mehrere Forth-Versionen für den C 64 und suche eine Möglichkeit, Real-Zahlen zu implementieren. Wer weiß, wo ein solches Real-Vokabular erhältlich ist, oder wer hat selbst ein solches geschrieben?

Nach vielen Versuchen mit verschiedenen Forth-Versionen stellt sich mir die Frage, ob es in Forth überhaupt möglich ist, in hochauflösender Grafik zu arbeiten. Welche Forth-Version ist hierfür geeignet, und was muß man eingeben?

Alois Schneider

C 64 im IBM-Gehäuse?

Wo bekomme ich ein IBM-ähnliches Gehäuse für den Einbau des C 64, Netzteil, Interface und Floppy?

Stefan Ullmann

Fragen Sie doch!

Selbst bei sorgfältiger Lektüre von Handbüchern und Programmbeschreibungen bleiben beim Anwender immer wieder Fragen offen. Viel mehr Fragen ergeben sich bei Computer-Interessenten, die noch keine festen Kontakte zu Händlern, Herstellern oder Computerclubs haben. Sie können der Redaktion Ihre Fragen schreiben oder Probleme schildern (am einfachsten auf der Karte »Lesermeinung«). Wir veranlassen, daß sie von einem Fachmann beantwortet werden. Allgemein interessierende Fragen und Antworten werden im Rahmen des Leserforums veröffentlicht.

Unvermeidbare Garbage Collection?

Ich habe ein Tabellenwerk mit zirka 1000 Zeilen aufgebaut und gebe es formatiert (mit RIGHT$ und so weiter) auf dem Bildschirm beziehungsweise Drucker aus. Leider benutzt der C 64 zu jedem neuen Zeilenaufbau wegen des RIGHT$ jedesmal freien Speicherplatz. Da kann man über die Zellen 51,52 zusehen, wie der freie Bereich »verbraten« wird und die Garbage Collection dann (fürchterlich!) zuschlägt. Meine Frage: Wie kann man das Betriebssystem dazu zwingen, zum Beispiel in Schleifen immer denselben Platz zu benutzen? Die ausgefallensten POKEs wären mir gerade recht, da ich der Ansicht bin: »Das Herrchen bin ich«

Hans Peter Kastner

Die Lösung des Problems werden Sie sicherlich nach intensivem Studium unseres neuen Kurses über Strings (und Programmierung im allgemeinen) finden.

Umleitung?

Anläßlich eines Kaufs einer Schreibmaschine/Printer mit RS232-Anschluß ist folgende Frage aufgetaucht:

Da ich sehr häufig umfangreiche Maschinensprache-Programme verwende, in denen eine Ausgabe auf Drucker vorgesehen ist (zum Beispiel Vizawrite, Wordpro, Multiplan und andere), ergibt sich die Notwendigkeit, sämtliche Ausgaben, die für die Geräteadresse 4 vorgesehen sind, auf Adresse 2 (RS232-Port) umzuleiten. Denn was nützt mir das beste Textverarbeitungssystem, wenn die Ausgabe auf meiner Schreibmaschine nicht möglich ist? Daher meine Frage an die Maschinensprache-Experten: Gibt es eine Möglichkeit der Umleitung von Adresse 4 auf 2, eventuell durch Ändern der Vektoren für die Routinen CHKOUT, CHROUT oder Ähnliche? Wie finde ich die Stellen im Programm, die umgeschrieben werden müssen?

Michael Fiedler

Grafikprobleme mit dem VC 20?

Ist das Grafik-Subsysteme der Graf-Elektronik-Systeme GmbH, Kempten an den VC 20, eventuell über VC 1011 B (Interface RS232), anschließbar? Wer hat einschlägige Erfahrungen?

Friedrich Dormeier

C 64 extern anhalten

Wie kann man den C 64 über externe Beschaltung beliebig anhalten und wieder starten?
Ausgabe 10/84, Ulrich Lang

Das Anhalten des C 64 ist theoretisch sehr einfach möglich, da es computerintern andauernd durchgeführt wird. Hier ist es der VIC, der zur Bildaufbereitung mehr Zeit benötigt, als es der Systemtakt erlaubt. Zu diesem Zweck hält er den Prozessor alle paar Millisekunden an, um die nötige Zeit zu bekommen. Die Leitung am Prozessor, die dies ermöglicht, ist das RDY-Pin. Wird dieses auf LOW-Pegel gehalten, so stoppt der Prozessor beim nächsten Lesezyklus und fährt erst bei RDY = 1 mit der Arbeit fort. Sie müßten also nur einen Zugriff auf dieses Pin durchführen, um das gewünschte Resultat zu erreichen. Wie Sie dem Handbuch entnehmen können, ist am Expansion-Port eine Leitung mit der Bezeichnung »BA« herausgeführt. Diese stellt das schon erwähnte Stopp-Signal vom VIC dar und ist direkt mit der RDY-Leitung des Prozessors verbunden.

Karsten Schramm

List-Stop kollidiert mit DOS 5.1

Das Programm »List-Stop« von Manfred Selke, Ausgabe 9/84, Seite 97, benutzt die »←«-Taste, die auch das DOS 5.1 für sich beansprucht. Welche Änderung ist nötig, um ein anderes Zeichen zu wählen?
Heinz Wagner, in Vertretung für viele andere

Die Pfeil-links-Taste hat den CHR$-Code 95. Sehen Sie im C 64-Handbuch auf Seite 136 nach. Wenn Sie in Zeile 112 des Listings die Zahl 95 in eine andere ändern, ist dieses Problem schon gelöst. Für zum Beispiel das »Pfund-Zeichen« ändern Sie in 92, für die F1-Taste in 133 um.

Der C 64 an der Stereoanlage

Wie schließe ich meinen C 64 an eine Stereoanlage an?

Kann man Maschinensprache-programme im Diskettenpuffer ablegen und dort laufen lassen?
Ausgabe 10/84 Thomas Denner

Der Anschluß des C 64 an eine Stereoanlage ist kein Problem und erfolgt am besten über den TAPE-Eingang. Stecken Sie ein Stereo-DIN-Kabel ein, so wie es ist, dann hören Sie den Ton nur auf einem Kanal. Um den Ton ohne Umschalter auf beide Kanäle zu bringen, müssen Sie den Stecker des Kabels öffnen und die Anschlüsse 3 und 5 miteinander verbinden (den Anschlußplan des Steckers können Sie aus dem Commodore-Handbuch Seite 142 ableiten). Danach sollten Sie alle Leitungen, bis auf die Erdung (2) kappen, um Störungen zu vermeiden. Probieren Sie das Kabel jetzt, so müßte der Ton über beide Lautsprecher der Anlage kommen.

Ohne weiteres ist es möglich, Programme in die Puffer der Floppy zu schreiben und sie dort auszuführen. Verwendung finden dabei die Befehle:

M-W ADL ADH Anzahl Data1 Data2 ...
M-E ADL ADH

Die Werte werden hierbei als CHR$-Codes übergeben; sie bedeuten:

ADL-LO-Byte der Adresse
ADH-HI-Byte der Adresse
Anzahl-Anzahl der Bytes
Data-Werte, die geschrieben werden sollen

Ich möchte hierbei auf den Floppykurs verweisen, der in der Ausgabe 10 begann; er beschäftigt sich mit solchen Problemen.

Karsten Schramm

Vielsaitig für C 64

Gibt es das Programm »Vielsaitig« auf für den C 64?
Ausgabe 10/84

Oliver Kreuzahler

Ich bin der Autor des Programms »Vielsaitig« für den VC 20. Es gibt jetzt auch eine sehr viel umfangreichere Version für den C 64 unter dem Namen »Gitarre 64«. Das Programm liegt ebenso wie mein Programm »Synthesizer 64« (Test in Happy Computer, Ausgabe 11/84) seit einem Jahr bei der Firma Commodore, Frankfurt in der Schublade und ich zweifle noch, ob es bei den dortigen Verhältnissen in der Software-Abteilung je in den Handel kommen wird. Da ich durch einen Vertrag gebunden bin, kann ich die Programme nicht anderweitig vertreiben lassen. Interessenten sollten also bei ihrem Computerhändler nachfragen, ob das Programm schon erschienen ist.

Ansonsten sollten Sie sich an mich wenden, gerade wenn Sie an der neuesten Version interessiert sind. Meine Adresse:
Werner Kracht, Espellohweg 38,
2000 Hamburg 52.

Drucker-Test: Itoh 8510

Ausgabe 11/84, Seite 22

Der Test über den Drucker Itoh 8510 schließt mit einem sehr guten Gesamtergebnis, das leider durch einen kleinen Fehler getrübt zu sein scheint. Auf Seite 161 bemängelt Herr Wängler, daß einzig eine gedehnte Schrift fehlt. Dem ist aber nicht so. Die Steuerzeichen für Breitschrift ein beziehungsweise aus sind CHR$(14) beziehungsweise CHR $(15).

Die Besprechung einzelner Artikel der verschiedenen Zeitschriften Ihres Verlages ist fester Bestandteil in unseren Clubabenden. Für die 64'er Zeitschrift können wir Ihnen das Testurteil »sehr gut, besonders empfehlenswert« ausstellen.

Peter Koch, 1. Vorsitzender, Computer-Club Bruchsal e.V., Spöckweg 27, 7520 Bruchsal

Zunächst einmal Dank für die Ergänzung zu dem Drucker-Test und für das ausgesprochene Lob.

Dies möchten wir noch einmal kurz zum Anlaß nehmen, um uns bei all den Lesern zu bedanken, die durch ihre rege Mitarbeit an der inhaltlichen Gestaltung geholfen haben, das 64'er zu dem zu machen was es heute ist. Nämlich eine Fachzeitschrift, aus der Heimcomputer-Anwender viele nützliche Tips, Anregungen, Programme und Entscheidungshilfen beziehen können.

Es soll aber auch als Anregung dienen, weiterhin die Redaktion tat- und schreibkräftig zu unterstützen.

Wollen Sie antworten?

Wir veröffentlichen auf dieser Seite auch Fragen, die sich nicht ohne weiteres anhand eines guten Archivs oder aufgrund der Sachkunde eines Herstellers beziehungsweise Programmierers beantworten lassen. Das ist vor allem der Fall, wenn es um bestimmte Erfahrungen geht oder um die Suche nach speziellen Programmen. Wenn Sie eine Antwort auf eine hier veröffentlichte Frage wissen — oder eine andere, bessere Antwort als die hier gelesene, dann schreiben Sie uns. Antworten publizieren wir in einer der nächsten Ausgaben. Bei Bedarf stellen wir auch den Kontakt zwischen Lesern her.

Textverarbeitung mit Seikosha GP-700A

Wie kann ich meinen Text durch Einfügen von Formatierungsbefehlen farbig gestalten?

Ausgabe 10/84

Conny Scharfenberg

Das Interface Typ 9200 ist ein Universaltyp, mit dem die verschiedensten Drucker an den C 64 angeschlossen werden können. Da praktisch alle Druckertypen unterschiedliche Steuercodes besitzen, ist dieses Interface nicht auf einen speziellen Typ angepaßt. Speziell für die Verwendung mit dem Farbdrucker GP-700A liefert Data Becker ein Interface, das auf diesen Drucker angepaßt ist und daher wesentlich mehr Funktionen bietet.

Reinhard Wiesemann



Horizontales Scrolling

Wie kann man beim C 64 ein horizontales Fine-Scrolling auf der Textseite simulieren?

Ausgabe 10/84

Christoph Bergmann

Die Voraussetzung für mein kleines Demo-Programm bietet das VIC-Register 22, in dem das kontinuierliche Setzen der ersten 3 Bits eine Verschiebung der gesamten Textseite um jeweils einen Punkt ermöglicht, wenn der Wert mit 248 geordert wird. Werden auf diese Weise insgesamt 7 Bits verschoben, so folgt darauf die Verschiebung um ein Byte, was durch die Basic-Funktion CHR$(20) simuliert wird.

Startet man das Demo-Programm, so kann es vorkommen, daß der einzeilige Text beim Verschieben flackert, was darauf zurückzuführen ist, daß die Verschiebung der Textzeile mit CHR$(20) einfach zu langsam ist. Abhilfe bietet hier eine Maschinensprachroutine, die das Byte-für-Byte-Scrolling übernehmen müßte. Hier nun das kleine Programm:

```
1 PRINT CHR$(147)
2 FOR A=1TO8:PRINT:NEXTA
3 PRINT "64'ER MAGAZIN"
4 PRINT CHR$(145),,
5 VIC=53248
6 FOR A=1TO20
7POKEVIC+22,
(PEEK(VIC+22)AND248)OR7
8 PRINT CHR$(20);
9 FOR B=6TO0STEP-1
10POKEVIC+22,
(PEEK(VIC+22)AND248)ORB
11 FOR C=1TO10
12 NEXT C,B,A
READY
```

Martin Althaus

Hier gibt's Mailbox-nummern

Gesammelt von unserem Leser André Steden (EMS). Bei den hervorgehobenen Nummern bedient ein C 64 die Mailbox.

Nummer	Mailbox
0201/1833781	Uni Essen
0201/237399	**Schossau**
0201/274625	**EMS**
0202/448202	Mailbox Wuppertal
0202/448204	W.M.S.
0202/556136	Toelleturm
0203/782497	Mercator Mailbox
0208/401763	**RAF – Mühlheim**
0209/271666	Beate Vollrath Box
0211/328249	EDV
0211/414579	Software Express
0211/593453	Epson
02151/779243	Hawischa
02151/801339	**K.I.S.**
02161/200928	Symic
02202/50033	Computer Center
0221/1616284	Saturn
0221/236534	U.M.S.
0221/371076	WDR-Computerclub
02234/58603	F.I.S.
0231/170414	Dortmunder Mailbox
0231/650786	C.B.B.S.
02331/16401	**Kobra-Box Hagen**
0234/7004023	Uni Bochum
02361/72928	?
02366/38536	Data Voigt
02373/66877	Ueding Elektronik
02383/50866	IGS
0241/870555	**A.I.S.**
0281/65466	W.I.S.
02841/66241	Esprit
030/3052635	Berliner Mailbox
030/7115078	TIC
040/41233098	Uni Hamburg
040/4916117	H.I.S.
040/5246387	W-W-S
040/6523486	M.C.S.
040/7540598	C64 User Club
04101/23789	Wang Info
04348/7513	N.C.S.
06081/9677	Taunus Mailbox
06102/51775	Lammy Mailbox
06154/51433	Decates
06181/48884	Otis
06826/2234	Hobby Comp.1
06826/6344	Hobby Comp.2
069/816787	Tecos
069/835037	IBM PC
07031/278296	Elias
0721/682607	M.C.S. Karlsruhe
089/132535	Info-Control
089/222066	Graphton
089/2800310	Cyber
089/596422	Tedas I
089/598423	Tedas II
09363/5329	Mailhouse

Hier gibt's Clubs

Star-Computer
Schlüterstr. 6
2000 Hamburg 13
Mo.-Fr. 14.00-18.00 Uhr
Tel. 040/452090

C 64 Club Bramsche
Lutterdamm 13 a
4550 Bramsche 1

VC 20 Club
Rainer Plapst
Haydnstr. 39
8906 Gersthofen

Mit dem C 16 stellt Commodore nicht nur einen neuen Computer vor, sondern ein ganzes System (Bild 1, 6 und 7). Computer, Datasette und Joysticks präsentieren sich im schwarz-grauen »Profi-Look«, Drucker und Floppy-Laufwerk des VC 20/C 64-Systems passen auch an den C 16. Gleichzeitig mit der Markteinführung des neuen Gerätes ist auch erste Software — zunächst auf Kassette — erhältlich. Commodore trägt damit der Tatsache Rechnung, daß angesichts eines umkämpften Marktes ein neuer Computer ohne Peripherie und Softwareangebot kaum noch erfolgreich einführbar ist.

Um es vorweg zu sagen: C 16 und 116 sind hard- und softwaremäßig völlig identisch. Der einzige Unterschied besteht bei Gehäuse und Tastatur. Während der C 16 hier der durch VC 20 und C 64 vorgegebenen Linie folgt, ist der 116 mit seiner Radiergummi-Tastatur und dem Miniatur-Gehäuse wohl eher als direkter Angriff auf den Sinclair Spectrum zu werten.

Doch sehen wir uns zunächst den C 16 etwas genauer an. Ein erster Blick auf die Tastatur (Bild 2) offenbart schon einige Unterschiede zum VC 20/C 64. Aus den beiden Cursortasten des Vorgängers wurden beim C 16 deren vier, die aber leider etwas ungünstig rechts oben in einer Reihe in den Tastaturblock integriert wurden. Um für diese Anordnung Platz zu schaffen, mußten einige andere Tasten verlegt werden. So findet der an die VC 20-Tastatur gewöhnte Programmierer die häufig benötigten Tasten »+«, »-«, »*«, »↑« und »=« nicht mehr an ihrem gewohnten Platz, was zu Anfang recht lästig ist. Insbesondere die Anordnung der »=«-Taste ganz rechts unten ist sehr unglücklich gewählt.

Die unterste Funktionstaste ist jetzt mit »HELP« beschriftet und hat eine spezielle Bedeutung bei der Fehlersuche. Drückt man nämlich diese Taste, nachdem der Computer eine Fehlermeldung angezeigt hat, dann wird die fehlerhafte Zeile sofort aufgelistet und der Abschnitt, in dem der Fehler auftrat, blinkt mit der Cursorfrequenz. Eine feine Sache bei der Programmentwicklung, allerdings wäre es schön, wenn man das mit der Zeit doch etwas nervende Blinken auf irgendeine einfache Art abstellen könnte.

Eine Restore-Taste gibt es nicht mehr, die Linkspfeiltaste des VC 20 / C 64 ist jetzt mit »ESC« (mehr darüber später) beschriftet und im Vordergrund fallen zwei neu beschriftete Tasten, »FLASH ON« und »FLASH OFF«, ins Auge. Zusammen mit der CTRL-Taste wird dadurch ein Blink-Modus ein- beziehungsweise ausgeschaltet (analog zu RVS ON/OFF).

Die Funktionstasten sind mit Basic-Befehlen belegt. Sonst entspricht sowohl die Tastatur als auch der Zeichensatz dem »Commodore-Standard«, man wird als »Umsteiger« wenig Schwierigkeiten haben.





Bild 2. Die Tastatur ist bewährt und gut. Deutlich erkennbar ist die gegenüber dem VC 20 geänderte Belegung einiger Tasten.



Bild 4. Viele Anschlußmöglichkeiten an der Rückseite (vor... buchse, Monitorausgang, serieller Port und Datasetten-Ans...



Bild 1. Das C 16-System

Ein erster Blick an die rechte Seite des C 16, wo man ganz richtig den Einschaltknopf vermutet, und sofort

Kompatibel um keinen Preis?

fallen zwei Dinge ins Auge (Bild 3). Als erstes, und zwar sehr positiv, ein kleiner Reset-Schalter — unverständlich, daß es Computer gibt, die keinen haben. Links daneben zwei winzige Buchsen, darüber steht etwas geschrieben. Man liest es, reibt sich die Augen, schaut nochmals hin — tatsächlich, es ist wahr: Die Mikro-Buchsen sind mit JOY 1 und JOY 2 beschriftet. Endlich einmal ein Computer, an den garantiert kein Joystick außer einem ganz speziellen Commodore-Stick mehr paßt. Diese neuen Joystickanschlüsse sollen über eine verbesserte Abschirmung verfügen, aber es bleibt die Frage, ob man den gleichen Effekt nicht auch mit Standard-Buchsen hätte erreichen können.

Was macht nun jemand, der zum Beispiel vom VC 20 auf den C 16 umsteigt, mit seinen vorhandenen Joysticks? Nun, vermutlich das gleiche wie mit seiner Datasette, denn auch der Datasettenanschluß wurde geändert. Da hilft alles nichts, entweder wird im Eigenbau ein entsprechender Zwischenstecker hergestellt, oder die alte Datasette wandert zusammen mit dem Lieblingsjoystick in eine Verkaufsanzeige. Eine dritte Möglichkeit: Auf ein Angebot aus dem Zubehörhandel warten. Ob so etwas aber gerade ein besonders gutes Argument ist, auf den C 16 umzusteigen, das mag dahingestellt bleiben.

Problemlos erweiterbar?

Ein ängstlicher Blick auf die Rückseite des Computers (Bild 4) zeigt, daß wenigstens der serielle Bus nicht mit Spezialbuchsen versehen wurde. Floppy und Drucker sind also weiterhin problemlos anzuschließen. Ein Video-Modulator ist wie beim C 64 fest eingebaut. Auffällig ist das Fehlen eines User-Ports, bisher Kennzeichen aller Commodore-Computer.

Der Expansion-Port dient zum Aufnehmen von Steckmodulen mit fertiger Software, sowie zum Anschluß einer (noch nicht erhältlichen) Speichererweiterung. Zu diesem Thema wäre zu bemerken, daß das Betriebssystem mit zwei Speicherbanks arbeitet. Zwischen den 32 KByte ROM von Betriebssystem und Basic und den (noch) 16 KByte RAM wird mittels Bank-Switching hin- und hergeschaltet. Dadurch kann man mit PEEK nicht ins ROM »hineinschauen«, sondern bewegt sich nur auf der RAM-Ebene. Nach Einbau einer 64-KByte-RAM-Erweiterung sollten daher tatsächlich fast 60 KByte für Basic-Programme zur Verfügung stehen.



nach rechts): Expansion-Port, Antennen-

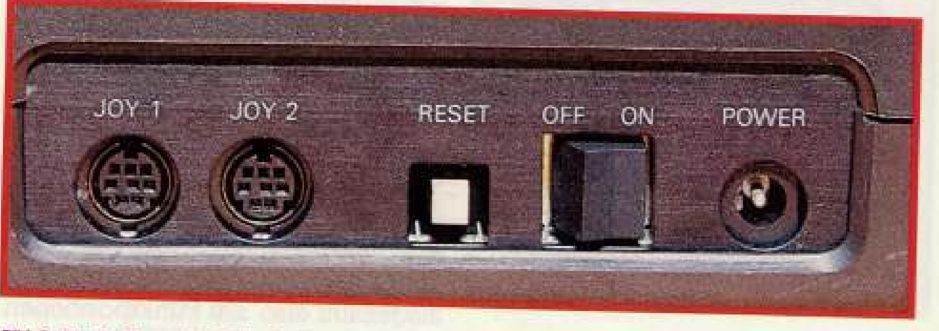


Bild 3. Der C 16 von der Seite. Deutlich zu erkennen der weiße Reset-Schalter und die Mikro-Joystickbuchsen. Auch der Netzteil-Anschluß wurde geändert.

Eine solche RAM-Erweiterung hätte übrigens bequem noch im Gehäuse des C 16 Platz. Ein Blick dort hinein auf die Platine (Bild 5) offenbart ein sehr aufgeräumtes Innenleben. Die großen integrierten Bausteine, insbesondere die neue CPU 7501 (kompatibel mit 6502/6510) und der ebenfalls neuentwickelte TED 7360, stellen Eigenentwicklungen von Commodore dar und werden nicht frei gehandelt. Informationen über diese Bausteine gibt es daher — ganz nach Art des Hauses — praktisch keine.

Auffällig ist das Fehlen weiterer Peripheriebausteine wie VIA oder CIA. Die Funktionen dieser Bausteine wurden in den TED integriert, der sich auch um die Video-Darstellung und die Tonerzeugung kümmert und so den Prozessor entlastet.

Betriebssystem und Basic sind in je einem 128 KBit ROM untergebracht, die 16 KByte RAM befinden sich in zwei TMS 4416-Chips. Zur besseren Wärmeableitung und Abschirmung ist über der Platine eine gelochte Metallplatte angebracht (im Bild 5 entfernt), von der ein Ausleger direkt mit dem scheinbar besonders kühlungsbedürftigen TED-Baustein Kontakt hat.

Besonders bemerkenswert: Bei unserem Testgerät waren alle hochintegrierten IC gesockelt. Es bleibt abzuwarten, ob diese gute Technik auch bei größeren Stückzahlen beibehalten wird. Und größere Stückzahlen werden von diesem Computer mit Sicherheit verkauft werden, dafür sorgt schon das bemerkenswert gute und umfangreiche Basic 3.5, das den C 16 zum idealen Computer für alle diejenigen macht, die wenig mit Maschinensprache im Sinn haben, aber trotzdem gute Programme schreiben wollen.



Bild 6. Die Datasette präsentiert sich auch im neuen Design

Gute Programme für C 64 und VC 20 zeichnen sich im wesentlichen dadurch aus, daß man sie nicht mehr lesen kann. Nach LIST flimmert dort, zumindest bei Programmen, die mit Grafik und Tonuntermalung arbeiten, ein unergründliches Gemisch von POKE, PEEK, SYS und sehr viel DATA über den Bildschirm, ab und zu auch einmal ein »normaler« Basic-Befehl (in der Regel ein »GOTO«). Wenn man sich einmal vor Augen hält, daß POKE, PEEK und SYS die Verbindung zwi-

Programmieren ohne POKEs

schen Basic und Maschinensprache herstellen, dann kann man ruhigen Gewissens sagen, daß die bisherigen Commodore-Heimcomputer im wesentlichen in Maschinensprache programmiert werden mußten — unverständlich für den Einsteiger, schwer erträglich aber auch für den Profi, der kostbare Programmierzeit damit vergeudete, sich seine eigene Basic-Erweiterung zu basteln, um überhaupt erst vernünftig arbeiten zu können.

Mit dem Basic 3.5 beschreitet Commodore jetzt ganz offensichtlich einen anderen Weg. Von der simplen Farbwahl über die Tonerzeugung bis hin zur hochauflösenden Grafik läßt sich alles mit entsprechend leistungsfähigen Basic-Befehlen programmieren. Daneben wird natürlich auch die strukturierte Programmierung unterstützt. Sprachkonstruktionen wie IF...THEN ...ELSE, DO WHILE oder DO UNTIL machen die dem Programm zugrunde liegende Idee im Listing sichtbar und vermeiden umständliche (und langsame) GOTO-Sprünge.

Natürlich ist Basic 3.5 vollständig aufwärtskompatibel zum altertümlichen V.2.0-Minimal-Basic des VC 20/C 64. Insgesamt ist das Basic 3.5 so leistungsfähig, daß alleine eine genaue Beschreibung aller Befehle und Funktionen leicht ein ganzes Sonderheft füllen würde (Tabelle 1). Beschränken wir uns daher auf die Betrachtung einiger wichtiger Aspekte.

Die Grafik

Der Bildschirm des C 16 hat eine Aufteilung von 25 Zeilen zu je 40 Zeichen. Jedes Zeichen wird in einer 8x8-Matrix dargestellt. Damit gibt es insgesamt also 320 Punktpositionen (40x8) pro Zeile, und 200 Punktpositionen (25x8) in vertikaler Richtung.

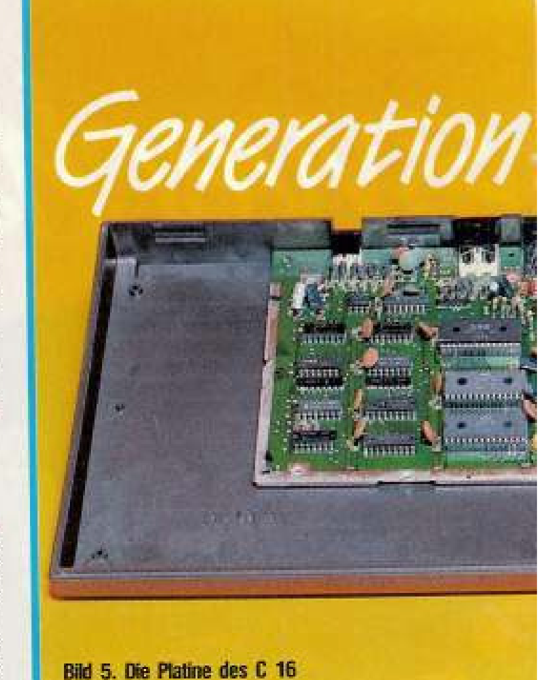


Bild 5. Die Platine des C 16

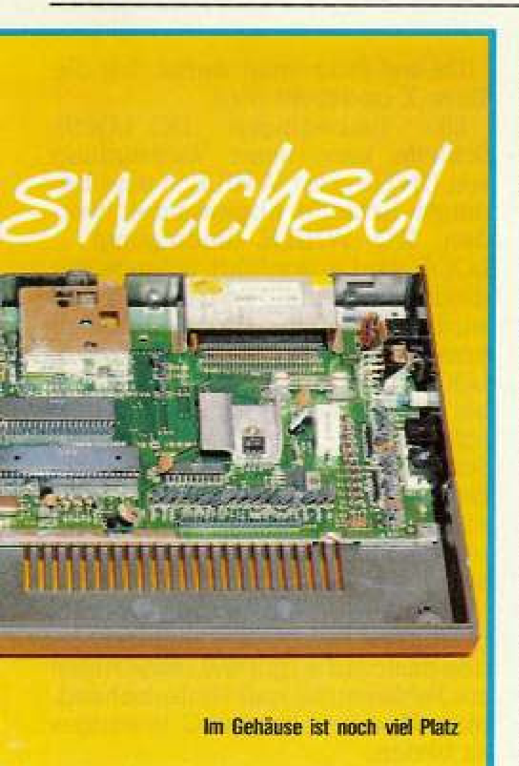
Genausoviele Punkte, nämlich 320 mal 200, können im hochauflösenden Grafik-Modus einzeln angesprochen werden. Mit insgesamt 64000 Einzelpunkten erreicht der C 16 damit die Grafik-Auflösung des C 64, was in etwa einer Verdopplung gegenüber dem VC 20 gleichkommt.

Daneben ist noch ein Mehrfarbenmodus mit halbierter Auflösung vorgesehen, bei dem jeder der dann 32000 Einzelpunkte eine von vier möglichen Farben annehmen kann.

Bis hier herrscht noch eine völlige Übereinstimmung zum Grafik-Konzept des C 64. Der große Unterschied ist nun, daß die C 16-Grafik vom Basic voll unterstützt wird.

Um beispielsweise mit dem C 64 (ohne Erweiterung!) einen Kreis oder eine Ellipse in hochauflösender Grafik zu zeichnen, und zusätzlich noch einen Text einzublenden, benötigt man neben fundierten Kenntnissen über die interne Organisation seines Computers und einigen Jahren Programmiererfahrung mindestens einige Dutzend Basic-Zeilen und jede Menge Geduld — denn Basic-Grafik ist langsam, wenn die primitivsten Dinge mühselig simuliert werden müssen. So werden ganze Kurse und Bücher damit gefüllt, auf dem C 64 das zu erreichen, wofür man beim C 16 drei Basic-Befehle und — als absoluter Neuling — maximal einige Minuten blättern im Handbuch braucht:

```
10 GRAPHIC 1,1 : CIRCLE, 160,100,
60,30 : CHAR,18,12, »HALLO«
```



Im Gehäuse ist noch viel Platz

Mit GRAPHIC 1,1 wird in den hochauflösenden Grafikmodus geschaltet. Der erste Parameter gibt den gewählten Modus an (0 für Text, 1 für Hochauflösung, 2 für Hochauflösung mit Textfenster, 3 für Mehrfarben-Modus, 4 für Mehrfarben-Modus mit Textfenster). Als Textfenster sind dabei die untersten fünf Bildschirmzeilen vorgesehen. Dieses Textfenster ist dann mit normalen PRINT-Befehlen ansprechbar. Will man Text direkt in die hochauflösende Grafik hineinmischen, dann bedient man sich des CHAR-Befehls. Der erste Parameter bestimmt im Mehrfarbenmodus die Textfarbe (und wird daher in unserem Beispiel für hochauflösende Grafik durch ein einzelnes Komma ersetzt). Die beiden folgenden Parameter kennzeichnen die Cursorposition, an welcher der Text ausgegeben werden soll. Als letztes muß noch ein Textstring angegeben werden, der dann ab der spezifizierten Position in die Grafik eingeblendet wird. Ein fünfter Parameter ist optional, nämlich die Angabe, ob der Text normal oder revers erscheinen soll.

Bleibt nur noch der CIRCLE-Befehl, der mit bis zu neun Parametern sehr komplex sein kann. Der erste Parameter gibt wieder die Farbzone an und ist nur im Mehrfarbenmodus zu verwenden. Dann folgen die Mittelpunktkoordinaten und der Radius in X- und Y-Richtung. Die letzten beiden Werte sind bei einem Kreis natürlich gleich groß, und daher reicht es, nur den ersten davon anzugeben. In unserem kleinen Beispiel haben wir jedoch eine Ellipse gezeichnet, und zwar mit Mittelpunkt in (160,100) und den Halbachsenabmessungen 60 beziehungsweise 30 Punkte.

Aber der CIRCLE-Befehl ist noch weitaus leistungsfähiger. Weitere Parameter regeln das Zeichnen nur einzelner Segmente sowie eine Drehung der ganzen Figur um einen beliebigen Winkel. Neben dem Zeichnen von Kreisen und Ellipsen kann der CIRCLE-Befehl auch für beliebige Vielecke verwendet werden.

Daneben steht noch eine Anzahl weiterer leistungsfähiger Grafik-Befehle zur Verfügung. DRAW zeichnet Einzelpunkte oder Linien, BOX zaubert blitzschnell alle möglichen Rechtecke auf den Bildschirm. LOCATE dient zum Positionieren des Grafikcursors, mit PAINT werden geschlossene Flächen ausgefüllt. SCALE schließlich dient zur Skalierung der Zeichenfläche und SCNCLR löscht den Bildschirm unabhängig vom eingestellten Grafikmodus.

Shapes statt Sprites

Im Gegensatz zum C 64 sind beim C 16 keine Sprites vorgesehen. Dafür gibt es jedoch leistungsstarke Basic-Befehle, um Bildausschnitte aus der hochauflösenden Grafik — sogenannte »Shapes« — in Stringvariable abzuspeichern oder wieder auf den Bildschirm zu bringen. Zum Beispiel wird mit »SSHAPE A$,100, 100,120,130« der Inhalt des Rechtecks mit linker oberer Ecke (100,100) und rechter unterer Ecke (120,130) auf dem Grafikbildschirm in der Stringvariablen A$ abgelegt. Mit »GSHAPE A$, 60,70« wird die Grafikinformation aus A$ wieder als Rechteck mit linker oberer Ecke (60,70) abgelegt. Über einen zusätzlichen (optionalen) Parameter kann der Wiedergabemodus bestimmt werden: Shapes können genauso wie aufgenommen auch wieder eingeblendet werden (und überschreiben dabei den Hintergrund), sie können revers dargestellt und schließlich wahlweise auch ODER, UND oder EXKLUSIV-ODER mit dem Hintergrund verknüpft werden.

Mit diesen Shapes eröffnen sich natürlich speziell bei der Spieleprogrammierung ungeahnte Möglichkeiten. Basic-Spiele in hochauflösender Grafik sind keine Utopie mehr. Unter diesem Gesichtspunkt verzichtet man gerne auf ein paar kümmerliche Sprites, die sich über einem Blockgrafik-Hintergrund bewegen. Dies um so leichter, als sich Sprites mit dem Shape-Konzept leicht simulieren lassen: Man arbeitet einfach mit zwei Stringvariablen. Die eine enthält die Spielfigur, die andere den zugehörigen Hintergrund (der ja durch die Figur verdeckt ist). Durch wechselseitiges Laden und Speichern von Hintergrund und Spielfigur lassen sich sehr einfach die entsprechenden Bewegungseffekte erzielen — und das sowohl in hochauflösender Grafik als auch in Mehrfarbengrafik.

Farbe und Sound

Der C 16 hat eine Grundpalette von 16 Farben zur Verfügung, von denen jede (bis auf Schwarz) noch in acht verschiedenen Intensitätsstufen dargestellt werden kann. Das ergibt insgesamt eine beachtliche Auswahl von 121 Farbtönen. Wer bietet mehr für 398 Mark?

Mit dem COLOR-Kommando können dabei die Farben für Bildschirmrahmen, Hintergrund, Mehrfarbenmodus und auch die Zeichenfarbe gewählt werden.

Zur Tonerzeugung stehen zwei unabhängige Tongeneratoren zur Verfügung, von denen einer auch für Geräuscheffekte eingesetzt werden kann. POKE-Befehle sind auch hier nicht nötig. Mit dem SOUND-Befehl werden sowohl der gewünschte Tongenerator angewählt als auch Tonhöhe (Notenwert) und Klangdauer angegeben, mit VOL wird die Lautstärke eingestellt.



Bild 7. Nur dieser Commodore-Joystick kann derzeit angeschlossen werden

Das Besondere dabei ist, daß bis zu zwei SOUND-Befehle (einer pro Kanal) parallel zum Basic-Programm ausgeführt werden. Der Programmablauf wird durch einen SOUND-Befehl also nicht etwa aufgehalten, bis der Ton zu Ende gespielt wurde; das Programm läuft normal weiter, während der Ton entsprechend der gewählten Tondauer erklingt. Man kann sich wohl vorstellen, wie diese Fähigkeit ein Programm beschleunigt, wenn man sich vor Augen hält, daß Computer der Vorgängergeneration (VC 20/C 64) die Tondauer über leere FOR...NEXT-Schleifen bestimmten.

Komfortable Programmierung

Bei der Entwicklung von eigenen Programmen und der in der Regel notwendigen Fehlersuche kommen die eingebauten Programmierhilfen des 3.5 Basic erst richtig zur Geltung. Eine Reihe von Befehlen und speziellen Funktionen stehen zur Verfügung, die bisher bei Commodore-Computern unter der Abkürzung OGNV (oft gebraucht, nie vorhanden) liefen.

Eine automatische Zeilennumerierung mittels AUTO ist ebenso selbstverständlich wie ein RENUMBER-Befehl zum Neunumerieren des Programms (wobei wahlweise auch nur Programmteile numeriert werden können und der Zeilenabstand sowie die Startzeile natürlich frei wählbar sind).

Diskettenkommandos, die man früher umständlich mit »OPEN 1,8,15...« an die Floppy senden mußte, sind jetzt als Basic-Kommandos integriert. SCRATCH löscht beispielsweise ein File auf der Diskette. Das Laden und Speichern von Diskettenprogrammen geschieht jetzt mit DLOAD beziehungsweise DSAVE. DIRECTORY holt das Inhaltsverzeichnis der Diskette auf den Bildschirm, selbstverständlich ohne Programmverlust.

Für das häufig benötigte Warten auf einen Tastendruck gibt es den Spezialbefehl GETKEY A$, wodurch man sich das lästige »IF A$="" THEN...« spart.

Hinter RESTORE kann eine Zeilennummer angegeben werden, was das einfache Hantieren mit mehreren unabhängigen DATA-Blöcken erlaubt.

Formatierte Ausgabe ist mit PRINT USING möglich. Die dabei verwendeten Zeichen lassen sich mit dem PUDEF-Kommando umdefinieren. Beispielsweise kann man für die formatierte Ausgabe den (amerikanischen) Dezimalpunkt durch das in Europa übliche Komma ersetzen. Mit ZONE kann die Weite der TAB-Bereiche geändert werden.

Die häufig gebrauchte INSTR-Funktion (hat als Ergebnis die Position eines Teilstrings in einem anderen String) ist ebenso vorhanden wie die Funktion JOY zur einfachen Joystickabfrage.

Bemerkenswert ist auch, daß die MID$-Funktion jetzt auch auf der linken Seite einer Wertzuweisung stehen kann. Sei beispielsweise A$="HALLO". Nach Ausführung des Befehls »MID$(A$,2,1) = "E"« ist A$ dann gleich der Zeichenfolge »HELLO«.

Für die Umrechnung zwischen Dezimal und Hexadezimal sind die beiden Funktionen DEC und HEX$ vorhanden.

Strukturierte Programmierung

Basic 3.5-Programme sind in der Regel um einiges übersichtlicher (und dabei schneller) als VC 20/C 64-Programme. Der Grund ist einleuchtend: Durch zusätzliche Schleifenbefehle werden GOTO-Anweisungen eingespart, und damit entfällt auch die Suchzeit, um die Zeilennummer zu finden.

Daneben wurde auch die IF-Anweisung um die ELSE-Klausel erweitert, was die Programmierung in vielen Fällen vereinfacht.

Der Kern der neuen Schleifenstruktur besteht aus den Anweisungen DO und LOOP. Ähnlich wie FOR...NEXT umklammert DO... LOOP einen Programmteil. Die Wirkung ist die folgende: Bei Erreichen eines DO merkt sich der Basic-Interpreter die Adresse dieses DO-Befehls als Schleifenanfang. Wird dann im weiteren Verlauf das zugehörige LOOP gefunden, erfolgt sofort ein Rücksprung zur Position des DO-Befehls. Das ergibt eine »unendliche« Schleife, was allerdings in den meisten Fällen nicht erwünscht ist. Daher ist die EXIT-Anweisung vorgesehen, die ein Verlassen der Schleife und eine Fortsetzung des normalen Programmablaufs hinter dem LOOP-Befehl ermöglicht. In der Regel wird man das EXIT von einer bestimmten Bedingung abhängig machen. Beispiel:

```
10 DO
20 GET A$ : IF A$=»X« THEN EXIT
30 LOOP
```

Dieses Programm wartet, bis die Taste X gedrückt wird.

Die (unbedingte) DO...LOOP-Schleife kann unter Verwendung von UNTIL oder WHILE in eine bedingte Schleife abgewandelt werden. DO WHILE (Bedingung) ... LOOP wird ausgeführt, solange die (Bedingung) erfüllt ist. Durch UNTIL wird praktisch der umgekehrte Fall erzeugt. DO UNTIL (Bedingung) ... LOOP wird solange durchlaufen, bis die Bedingung erfüllt ist. Natürlich können auch bei bedingten Schleifen zusätzliche EXITs eingebaut werden. Das ermöglicht sehr effiziente Programme, insbesondere, wenn mehrere Bedingungen beachtet werden müssen.

Gemäß der Parole »wer viel programmiert macht viele Fehler« ist jedes Basic nur so gut wie seine Hilfen zur Fehlersuche und Fehlerbehandlung. Und hier hat der C 16 einiges zu bieten.

Die HELP-Funktion wurde bereits anfangs erwähnt und ermöglicht die schnelle Lokalisierung eines Fehlers innerhalb einer Programmzeile.

Für den nicht seltenen Fall, daß keine Fehlermeldung erfolgt, das Programm jedoch unsinnige Sachen macht (also irgendwo noch ein logischer Fehler steckt) kann man mit TRON eine Trace-Funktion einschalten. Dabei wird die Zeilennummer der gerade abgearbeiteten Zeile angezeigt, wodurch man so manchem Fehler leichter auf die Spur kommen kann. TROFF schalten den Trace wieder ab.

Debugging leicht gemacht

Für die Fehlerbehandlung innerhalb des Programms ist der TRAP-Befehl vorgesehen. Zum Beispiel wird mit der Anweisung »TRAP 1000« beim Auftreten eines Fehlers das Programm nicht mit entsprechender Meldung abgebrochen, sondern es wird in eine Fehlerbehandlungsroutine gesprungen (hier ab



Bild 8. Der Gummitasten-Zwilling C 116

Generationswechsel

TEST C 16

Zeile 1000). Die Nummer der Zeile, in der der Fehler auftrat, wird dabei in der Systemvariablen EL gespeichert. Die Variable ER enthält die Fehlernummer, und ERR$ die Fehlermeldung im Klartext. Mit diesen Informationen kann man in der Fehlerbehandlungsroutine entsprechende Maßnahmen ergreifen und schließlich mit RESUME den normalen Programmablauf wieder aufnehmen lassen. Übrigens wird auch das Drücken der Stop-Taste mit TRAP abgefangen.

Window-Technik

Bei soviel Licht fällt gelegentlich auch ein Schatten. Die für den C 16/116 angekündigte moderne Window-Technik, also das Arbeiten mit verschiedenen Bildschirmfenstern, ist leider nicht in einer vollends überzeugenden Form implementiert.

Es kann überhaupt nur ein einziges Window erzeugt werden, und das nicht etwa per Basic-Befehl (wie man es an sich erwarten würde), sondern über eine ESC-Funktion. Damit kommen wir gleich zur Bedeutung der ESC-Taste auf der Tastatur.

Es gibt nämlich 26 ESC-Funktionen, die durch Drücken von ESC, gefolgt von einer Buchstabentaste, aufgerufen werden (Tabelle 2).

Um das eine mögliche Fenster zu erzeugen, muß man den Cursor in die linke obere Ecke des vorgesehenen Windows bringen, dann ESC T drücken, anschließend in die rechte untere Ecke fahren und ESC B betätigen. Dadurch ist das Fenster definiert. Alle Ein- oder Ausgaben spielen sich jetzt ausschließlich hier ab.

Durch zweimaliges Drücken der Home-Taste wird das Window wieder gelöscht.

So fortgeschritten das Konzept auch gegenüber dem VC 20/C 64 ist, es bleibt einiges zu wünschen übrig. Die Methode der Window-Definition ist viel zu umständlich und zu langsam, zumal ein Befehl zur direkten Cursorpositionierung nicht vorhanden ist. Das fällt um so schwerer ins Gewicht, als immer nur ein einziges Window definiert werden kann, was aber in der Regel nicht sehr sinnvoll ist. Wenn man den Bildschirm aber in verschiedene Bereiche aufteilen will, dann wirkt sich das ständige umständliche Definieren der Fenster doch zum einen auf die Programmlänge, zum anderen auf die Abarbeitungsgeschwindigkeit negativ aus.

Dennoch ist das Windowing ein Schritt in die richtige Richtung, hin zum benutzerfreundlichen Computer. Für eine übersichtliche Bildschirmaufteilung besteht jedenfalls in fast jeder Programmiersituation ein Bedarf. Wo man sich früher damit behalf, den gesamten Bildschirm bei jeder Veränderung neu aufzubauen, ist es jetzt möglich, nur den wirklich zu ändernden Bereich anzusprechen, ohne dabei die übrigen Informationen zu beeinflussen.

Maschinensprache-Monitor eingebaut

Für Maschinensprachefreunde — und solche, die es werden wollen — hält der C 16 noch einen ganz besonderen Leckerbissen parat. Er verfügt nämlich über einen fest im ROM vorhandenen Maschinensprache-Monitor, genannt TEDMON.

TEDMON ist genau genommen sogar mehr als nur ein Monitorprogramm für Maschinensprache. Er enthält nämlich einen Disassembler und auch einen kleinen Assembler. Maschinenprogramme können mit TEDMON sehr komfortabel entwickelt und anschließend als schnelle Unterroutinen von Basic aus aufgerufen werden. Tabelle 3 zeigt den TEDMON-Befehlssatz.

Fazit

Neben dem C 16 bietet Commodore noch den 116 an, also praktisch den gleichen Computer, nur im anderen Gehäuse (Bild 8). Der Preisunterschied zwischen den Geräten (real 398 Mark für den C 16, 348 Mark für den 116, der empfohlene Verkaufspreis liegt jeweils 50 Mark höher) ist so gering, daß wohl kaum jemand für 50 Mark die Nachteile der Gummitastatur beim 116 in Kauf nehmen wird (es sei denn, ein hartnäckiger Spectrum-Freund, aber für den ist das dann eh der falsche Computer).

abs	log
and	loop
asc	mid$
atn	monitor
auto	new
backup	next
box	not
char	on
chr$	open
circle	or
close	paint
clr	peek
cmd	poke
collect	pos
color	print
cont	print#
copy	printusing
cos	pudef
data	rclr
dec	rdot
def	read
delete	rem
dim	rename
directory	renumber
dload	restore
do	resume
draw	return
ds	rgr
ds$	right$
dsave	rlum
el	rnd
end	run
er	save
err$	scale
exp	scnclr
fn	scratch
for	sgn
fre	sin
get	sound
getkey	spc(
get#	sqr
gosub	sshape
goto	st
graphic	stop
gshape	str$
header	sys
hex$	tab(
if	tan
input	ti
input#	ti$
instr	trap
int	troff
joy	tron
key	until
left$	usr
len	val
let	verify
list	vol
load	wait
locate	while

Tabelle 1. der Leistungsfähige Befehlssatz des C 16/116. Die farbig unterlegten Befehle sind beim VC 20/C 64 nicht vorhanden.

(ESC) & Taste	Funktion
A	Automatisch einfügen
B	(Set Bottom) Fixiert an der gegenwärtigen CURSOR-Position die rechte, untere Fensterecke
C	(Clear auto insert) Hebt automatisch Einfügen auf
D	(Delete) Löscht eine Zeile an der CURSOR-Position
I	(Insert) Fügt eine Zeile an der CURSOR-Position ein
J	CURSOR wird an den Anfang der CURSOR-Positionszeile gesetzt
K	CURSOR wird an das Ende der CURSOR-Positionszeile gesetzt
L	Schaltet SCROLLING-Modus ein
M	Schaltet SCROLLING-Modus aus
N	Schaltet zur normalen Bildschirmgröße zurück und löscht den Bildschirm
O	(Off) Hebt Einfüge-, Anführungszeichen-, Reverse- und Blink-Modus wieder auf
P	Löscht Bildschirmzeile vom Anfang bis zur CURSOR-Position
Q	Löscht Bildschirmzeile ab der CURSOR-Position bis zum Ende
R	Verkleinert das Bildschirmformat und löscht den Bildschirm
T	(Set Top) Fixiert an der gegenwärtigen CURSOR-Position die linke, obere Ecke des Fensters
V	SCROLLEN des Bildschirminhalts nach oben
W	SCROLLEN des Bildschirminhalts nach unten
X	(Exit ESC) Befreit Sie aus dem ESCAPE-Modus nach versehentlicher Betätigung der (ESC)-Taste

Tabelle 2. Die ESC-Funktionen

A	Assemble	Wandelt ein Mnemonic (Klartext-Befehl) in den entsprechenden Maschinencode des 6502 beziehungsweise 7501
C	Compare	Vergleicht zwei Speicherbereiche und zeigt die Unterschiede
D	Disassemble	Wandelt Maschinencode in Mnemonics (Klartext)
F	Fill	Füllt einen Speicherbereich mit wählbarem Wert
G	Go	Startet Maschinenprogramm an angegebener Adresse
H	Hunt	Durchsucht einen Speicherbereich nach einem bestimmten Wert und zeigt alle Speicherplätze an, die diesen Wert enthalten
L	Load	Lädt ein Programm von Kassette oder Diskette
M	Memory	Zeigt alle Inhalte eines wählbaren Speicherbereichs in Hexdarstellung an
R	Register	Ausgabe der aktuellen Registerinhalte
S	Save	Speichert ein Programm auf Kassette oder Diskette
T	Transfer	Blockkopierbefehl, kopiert einen bestimmten Speicherbereich in einen anderen
V	Verify	Vergleicht ein Programm im Arbeitsspeicher mit einem auf Kassette oder Diskette
X	Exit	Zurück zu Basic
.	(Punkt)	Entspricht dem A (Assemble)
>	(größer als)	Ändert bis zu 8 Byte ab bestimmter Speicherstelle (nach M-Befehl)
;	(Semikolon)	Ändert die 7501-Registerinhalte (nach R-Befehl)

Tabelle 3. Die TEDMON-Befehle

Der mit 16 KByte zu kleine Anwenderspeicher (nach Einschalten der hochauflösenden Grafik bleiben noch exakt 2045 Byte zum Programmieren) dürfte bereits in naher Zukunft mit entsprechenden RAM-Modulen erweiterbar sein. Man sollte aber nicht vergessen, daß man bei der Leistungsstärke des C 16 Basic in 2 KByte etwa das gleiche an Grafik-Programm unterbringen kann wie beim C 64 in 8 KByte.

Der C 16 jedenfalls ist insbesondere vom Basic her in der Tat mindestens eine ganze Generation weiter als der VC 20 und der C 64. Durch sehr komfortable Programmierhilfen und ein umfangreiches, praxisnahes Basic ist er nicht nur der ideale Einsteiger-Computer; auch Profis fahren in der Regel lieber einen Rolls Royce, als daß sie wirklich jede Strecke zu Fuß gehen.

(ev)

Wir suchen die Anwendung des Monats

Anwendung des Monats, was ist das? Nun, Sie haben einen Commodore 64 oder einen VC 20 und versuchen diesen irgendwie sinnvoll einzusetzen. Unter einer sinnvollen Anwendung versteht die 64'er Redaktion alles, was beispielsweise Programme im häuslichen Bereich bewirken. Es kann sich dabei um die Berechnung der Benzinkosten für Ihren Wagen handeln, um ein eigenes Textverarbeitungsprogramm gehen, sich um die Verwaltung Ihrer Tiefkühltruhe drehen oder ein ausgeklügeltes Telefon- und Adreßregister sein.

Setzen Sie Ihren VC 20/C 64 mehr oder weniger beruflich ein? Auch, oder vor allem, das ist eine sinnvolle Anwendung. Sie führen die Lohn- und Gehaltsabrechnung, Ihre Lagerverwaltung, die Bestellungen auf einem Commodore-Heimcomputer durch? So spezielle Anwendungen wie die Berechnung der Statik von selbstgezimmerten Regalen, von Klimadiagrammen oder Vokabellernprogrammen für den Schulunterricht oder die Zinsberechnung bei Krediten sind ebenfalls Themen, die mehr als konkurrenzfähig sind.

Uns ist die Anwendung des Monats



wert.

Schreiben Sie uns, was Sie mit Ihrem Computer machen:

Redaktion 64'er, Aktion Anwendung des Monats, Hans-Pinsel-Str. 2, 8013 Haar bei München

Hardware-Interface ganz weich

Zusammen mit dem Epson-Software-Interface EC-64 erhält man eine exzellente Ansteuerungssoftware, die noch einiges mehr kann als herkömmliche Epson-Interfaces.

Die Bezeichnung »Soft-Interface« besagt, wie Sie sicher schon vermuten, daß die Treibersoftware nicht wie bei herkömmlichen Hardware-Interfaces in ROMs auf der Platine enthalten ist, sondern extern auf Diskette mitgeliefert wird. Die Verbindung zum Drucker wird dabei durch ein einfaches User-Port-Centronics-Kabel hergestellt.

Das hat natürlich alles seine Vor- und Nachteile. Einerseits muß man erst die Software nachladen, die dann auch noch Speicherplatz verbraucht. Andererseits muß man aber zugeben, daß mit diesem »Softinterface« eine äußerst preisgünstige Alternative zur bestehenden Masse der Hardwareinterfaces geboten wird. Außerdem bietet das EC-64 einige Besonderheiten, die andere Epson-Interfaces nicht haben.

Das Speicherplatzproblem

Die Routinen zum Ansteuern des Druckers benötigen zirka 1 KByte RAM — nicht viel, aber oft störend, wenn man Programme benutzen will, die vielleicht gerade diesen bestimmten Speicherbereich benutzen. Aus diesem Grunde wurde bei der Software darauf geachtet, daß die Ansteuerungsroutinen in verschiedene Speicherbereiche gelegt werden können. Am C 64 stehen vier verschiedene Bereiche zur Verfügung, die einen problemlosen Betrieb mit Simons Basic, Textomat und diversen anderen Programmen ermöglichen. Bei der VC 20-Version, die auf der selben Diskette gespeichert ist, stehen fünf verschiedene Bereiche zur Verfügung. Eine Speichererweiterung wird allerdings vorausgesetzt. Die Warmstartadressen für diese Routinen sind, falls aus Versehen die Schnittstelle mit Stop/Restore ausgeschaltet wurde, in der mitgelieferten Bedienungsanleitung enthalten.

Vielseitige Möglichkeiten

Im Gegensatz zu vielen Hardware-Interfaces, die nur eine Geräteadresse benutzen, werden von diesem Softinterface vier Geräteadressen benutzt. Die Ansteuerung geschieht wie auch bei den Commodore-Druckern über OPEN X,Y: PRINT#X,.... wobei X die logische Filenummer und Y die verwendete Geräteadresse ist. Belegt werden die Geräteadressen #4, #5, #6 und #7. Die Bedeutung der einzelnen Adressen wollen wir einmal genauer unter die Lupe nehmen.

Bei Ansprechen der Geräteadresse #4 können sämtliche Möglichkeiten der Epson-Drucker genutzt werden. Da der Zeichensatz des C 64 nicht mit dem Standard-ASCII-Zeichensatz übereinstimmt, werden hier die Codes entsprechend umgewandelt. Sämtliche Epson-Steuerzeichen für Auswahl der Schriftarten, Superscript, Subscript etc. können dabei benutzt werden.

Die Adresse #5 hat die gleiche Funktion wie die Geräteadresse #4, es findet jedoch keine Code-Umwandlung statt.

Die wohl interessanteste Adresse ist Geräteadresse #6: Wenn Sie diese Adresse benutzen, dann druckt der Epson-Drucker in Commodore-Schrift und ist in der Lage, alle Commodore-Steuerzeichen auszugeben. Haben Sie auf Ihrem Bildschirm einen anderen Zeichensatz, so spuckt der Drucker seine Texte wie selbstverständlich mit diesem Zeichensatz aus — gleichgültig, in welchem Speicherbereich dieser liegt.

Die Geräteadresse #7 eröffnet dem Benutzer eine Möglichkeit, die eine Hardwareschnittstelle praktisch nie bieten kann (außer wenn Zusatzsoftware wie beim Print 64 geliefert wird). Gibt man über diese Adresse einen CHR$(0) aus, fängt der Epson-Drucker an, eine Hardcopy des Bildschirms zu drucken. Man könnte meinen, der Computer würde auf dem Bildschirm nachsehen, was er auf den Drucker ausgeben soll: Die Hardcopy sieht immer so aus wie der Bildschirminhalt, gleichgültig ob Multicolormodus, geänderter Zeichensatz, HiRes oder sonst irgend etwas. Das einzige, was nicht ganz einwandfrei ausgedruckt wird, sind Bilder, die mit dem Koalapad gemalt wurden. Das liegt aber wohl mehr an der etwas seltsamen Farbspeicherbenutzung des Koalapainters und nicht am Interface.

Das Senden eines CHR$(1) an den Drucker bei Geräteadresse #7 erzeugt eine invertierte Hardcopy. Wird über diese Adresse ein anderes Zeichen als CHR$(0) oder CHR$(1) gesendet, wird gedruckt, als würde Geräteadresse #5 verwendet.

Reißfeste Verbindung

Die »Hardware« besteht, wie schon erwähnt, aus einem Verbindungskabel zwischen dem User-Port des C 64 oder VC 20 und dem Centronics-Anschluß der Epson-Drucker RX-80 oder FX-80 beziehungsweise FX-100. Lobenswert ist dabei, daß nicht die billigsten Anschlußstecker und -kabel verwendet wurden, sondern wirklich stabiles Material die Ausgangsbasis für das EC-64 ist.

Ein tolles Ding

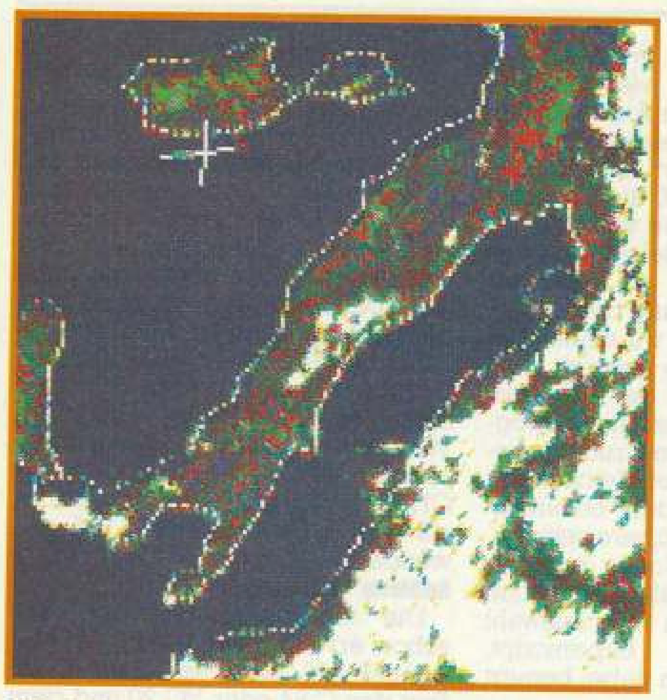
Das Epson EC-64-Softinterface ist eine der vielseitigsten und zugleich billigsten Möglichkeiten, einen Epson-Drucker an den C 64 oder VC 20 anzuschließen und sinnvoll zu nutzen. Besonders die Möglichkeit, eigene Zeichensätze und bildschirmgetreue Hardcopys auszudrucken, beeindruckt doch sehr, besonders im Hinblick auf die einfache Bedienung. Störend war nur, daß der Commodore-Zeichensatz auf Geräteadresse #6 liegt und deshalb Programme, die Commodore-Zeichensatz ausdrucken sollen, von Geräteadresse #4 auf #6 umgeschrieben werden müssen.

(M. Kohlen/aa)

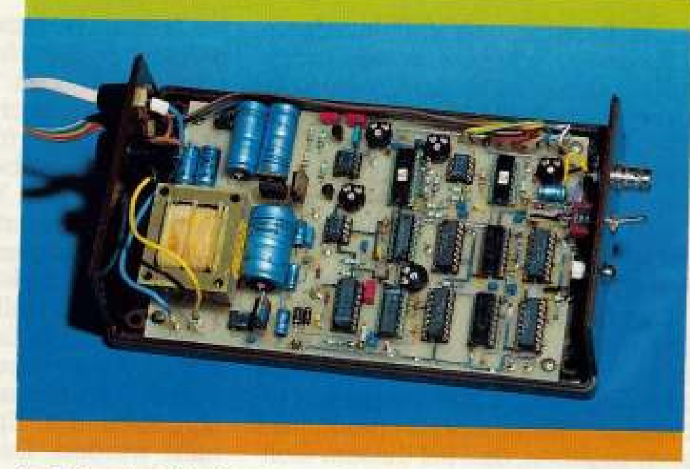
Info: Epson Deutschland, Am Seestern 24, 4000 Düsseldorf 4, Tel. (0211) 59520, Preis: 128 Mark.

Digitalisierte Bilder mit dem C64

Eine interessante Entwicklung für den C 64 sind die Videodigitalisierer. Diese Geräte gestatten, Videobilder in digitale Signale umzuwandeln, um sie auf dem Bildschirm oder dem Drucker in Schwarzweiß oder Farbe auszugeben.



Selbst digitalisierte Bilder von Satellitenaufnahmen können mit einem Farbdrucker ausgegeben werden



Der Aufbau des Videodigitalisierers überzeugt durch kompakte Bauweise

Einer dieser Digitalisierer ist das Gerät von Print Technik, das als Bausatz etwa 460 Mark kostet. Als Signalquelle benötigt diese Erweiterung ein beliebiges Videosignal über eine BNC-Buchse. Es kann direkt von einer Videokamera, einem Videorecorder oder einem entsprechenden Ausgang des Fernsehgeräts kommen. Die einzige Voraussetzung ist, daß das Bild zirka fünf Sekunden zur Verfügung steht. Diese Zeit braucht der Digitalisierer, um das Eingangssignal in 64 000 Einzelinformationen zu zerlegen.

Die mitgelieferte Software bietet im Hauptmenü folgende Funktionen: Bild darstellen, Bild einlesen, Bild drucken, Bild von der Diskette laden und Bild auf der Diskette speichern. Fünf Sekunden nach dem Start des Einlesevorgangs erscheint das Bild auf dem Bildschirm. Ist man mit dem dargestellten Ausschnitt nicht zufrieden, kann man das Bild mit den Cursor-Steuertasten horizontal und vertikal verschieben. Zu Beginn erscheint das Bild in Schwarzweiß. Mit den Funktionstasten F1 bis F8 kann den vier verschiedenen Graustufen dann jeweils eine Farbe zugeordnet werden. Kommt man dann doch zu dem Schluß, daß eine Schwarzweiß-Darstellung günstiger wäre, genügt ein Druck auf die »$«-Taste und der Ursprungszustand ist wieder hergestellt.

Das Innenleben dieses Gerätes ist durch den Einsatz integrierter Schaltkreise recht einfach aufgebaut, jedoch setzt die kurze Bauanleitung Kenntnisse im Aufbau elektronischer Schaltungen voraus.

Die Einsatzgebiete eines Videodigitalisierer sind recht vielseitig. Sie können von der Auswertung von Satellitenbildern bis hin zur Alarmanlage reichen. Bei dem Einsatz als Alarmanlage kann man ein einmal eingelesenes »Sollbild« mit später aufgezeichneten Bildern durch den C 64 vergleichen und bei einer Veränderung ein Signal aussenden lassen.

Ein Ausdruck mit einem Farb- oder Schwarzweißdrucker ist vom Menü aus anwählbar. Hierbei kann man vor dem Druckvorgang den Druckertyp bestimmen.

Als vorrangiges Anwendungsgebiet der Digitalisierer bietet sich der technisch-wissenschaftliche Bereich an, wobei der C 64 mit diesem Zusatzgerät auch hier als preiswerter Computer eingesetzt werden kann.

(Rainer Schönrock/rg)

Speichertuning für VC 20

Es gibt Computer wie den Sharp MZ-700 oder die meisten CP/M-Systeme, die sich dem Benutzer nach dem Einschalten völlig nackt präsentieren, das heißt fast der gesamte Speicherbereich besteht aus freiem RAM. Lediglich ein kleines Betriebssystem im ROM sorgt für das Laden der eigentlichen Programmiersprache.

Dieses Prinzip der reinen RAM-Maschine fand auch bei der Konzeption der Speichererweiterungskarte MR 64 von Roßmöller Verwendung (Bild 1). Die Organisation ähnelt der des C 64, der ROM-Bereich des Computers wird nämlich mit RAM überlagert.

Diese Erweiterungskarte für den VC 20 unterscheidet sich also sowohl von der Verwaltung des Speichers, als auch von der Art des Einbaus von den anderen beiden, in Ausgabe 9/84 getesteten, 64 KByte-Erweiterungen. Sie wird nämlich nicht in den Expansionsport eingesteckt, sondern in den Computer eingebaut. Damit bleibt der Erweiterungsanschluß für Module frei.

Einfacher Einbau

Zum Einbau wird das Gehäuse (freilich erst nach Ablauf der Garantiefrist) geöffnet und die CPU mit Hilfe eines Schraubenziehers aus ihrem angestammten Sockel entfernt. An deren Stelle steckt man den Verbindungsstecker ein, der über ein Flachbandkabel mit der MR 64-Platine verbunden ist.

Die 6502-CPU wird danach einfach wieder in den neuen Sockel auf der Speicherplatine eingesteckt — kinderleicht. Wer sich jedoch nicht auf den Umgang mit ICs versteht, kann die CPU durch falsche Polung in den Tod schicken, denn weder in der etwas mageren Beschreibung noch auf der Platine ist die Polungsrichtung der CPU angegeben. Also habe ich die 6502-CPU in der Richtung eingesteckt, wie ich sie aus dem Sockel gehebelt habe, und die war glücklicherweise richtig.

Nach dem Einschalten zeigt die Initialisierungsanzeige 28159 freie Bytes an, welches die größtmögliche — herstellerseitig vorgesehene Ausbauversion — ist. In diesem Punkt ist die MR 64 also identisch mit den üblichen Speicherweiterungen.

Einzelne Speicherbereiche wie zum Beispiel 3, 8, 16 KByte oder Modulspeicher können über DIL-Schalter ein- beziehungsweise ausgeschaltet werden. Bei eingebauter Karte ist dies sicherlich ein diffiziles Unterfangen; Roßmöller schlägt deshalb vor, Schalter ins Computergehäuse einzubauen (gehören nicht zum Lieferumfang).

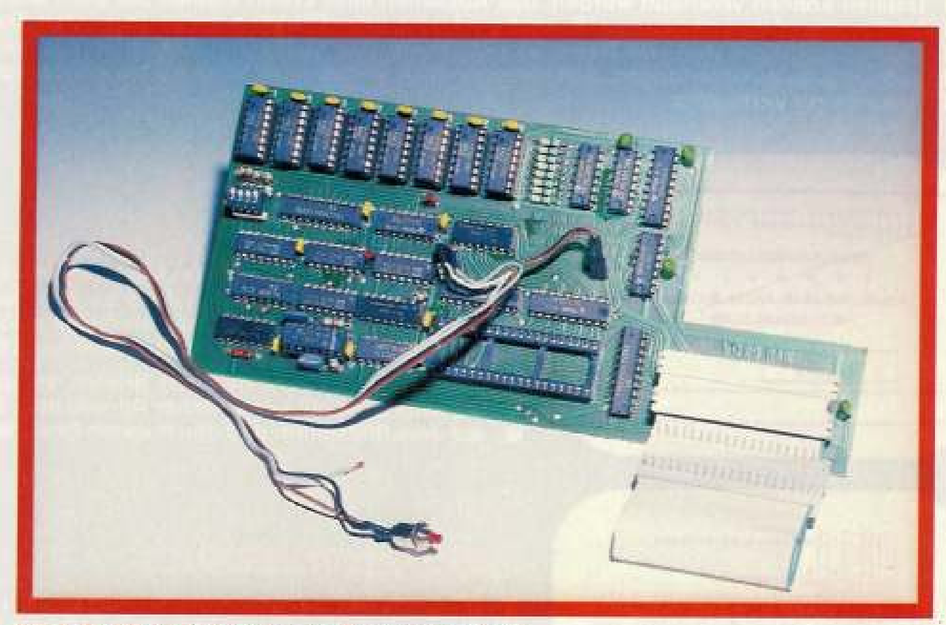


Bild 1. Die MR 64-Karte wird in den Computer eingebaut

Der Zugriff auf den Speicher

Wie — so wird sich mancher fragen — bewerkstelligt man den Zugriff auf andere Speicherbereiche?

Wie anfangs erwähnt, lehnt sich die Organisation des RAM an die des C 64 an. Damit kann man auch im ROM-Bereich Daten abspeichern, was Bild 2 verdeutlicht.

Über die Adresse 40959 (vergleichbar mit Speicherstelle 1 beim C 64) können bestimmte ROM-Abschnitte auf Schreib-/Lesespeicher softwaremäßig umgeschaltet werden. Im Umgang mit dieser Speicherstelle ist jedoch Vorsicht geboten, denn wird beispielsweise das Basic-ROM ausgeblendet, dann befindet sich an dieser Stelle nur noch leeres RAM. Dies hat in etwa den gleicher Effekt, wie das Absägen eines Astes, auf dem man gerade sitzt. Aus diesem Grund ist auch gleich ein Reset-Taster mit eingebaut worden.

Somit hat man also die Möglichkeit, den gesamten — vom Prozessor adressierbaren — Bereich auf RAM umzuschalten. Ohne erhebliche Erfahrung in Maschinensprache sollte man damit aber vorsichtig sein.

Zum Beispiel kann man den Basic-Interpreter und das Betriebssystem ins RAM kopieren und in diesem Bereich die Änderungen vornehmen. Dies kann über folgendes Programm geschehen:

```
FOR I= 49152 TO 65535: POKE I, PEEK (I): NEXT
```

Diese Zeile sieht auf Anhieb etwas seltsam aus, denn warum POKEt man in eine Speicherstelle einen Wert, der sich sowieso schon dort befindet? Ganz einfach. Mit PEEK

Eine neuartige 64 KByte-RAM-Karte für den VC 20 stellt 32 KByte freien Basic-Speicher zur Verfügung und ermöglicht Änderungen am Betriebssystem. Wird der VC 20 damit dem C 64 ebenbürtig?

liest man den Inhalt des ROMs aus und schreibt ihn mit POKE ins darunterliegende RAM. Da diese Übertragung in Basic sehr langsam ist (man muß ja immerhin 16 KByte kopieren) gehört zum Lieferumfang dieser Karte ein Maschinenprogramm, das diese Aufgabe in Sekunden erledigt.

Die gesamte Systemsoftware liegt nun im RAM und kann beliebig abgeändert werden. So kann beispielsweise der ASC-Befehl mit POKE 55183,5 entschärft werden, damit bei einem Leerstring ("") nicht »Illegal Quantity Error« — wie es normalerweise der Fall ist — sondern eine Null ausgegeben wird.

Natürlich sind auch weitergehende Ergänzungen und Änderungen auf diese Weise realisierbar. Lediglich beim Abspeichern gibt es Probleme, denn dieser Adreßbereich wird vom Betriebssystem nicht ohne weiteres auf Band aufgezeichnet. Commodore wollte damit verhindern, daß ein Spielmodul abgespeichert werden kann.

Diese Karte überlagert aber nicht nur die Systemsoftware ($C000 bis $FFFF) wie eben beschrieben, sondern auch den Zeichengenerator und den Ein-/Ausgabe-Bereich mit RAM.

Zusätzlicher Basic-Speicher

Mit dem 4 KByte-Zeichengenerator-ROM hat es eine besondere Bewandtnis. Wer denkt, daß man auch hier die Zeicheninformationen wie die Betriebssoftware überschreiben kann, der irrt. Aufgrund des eigenständigen Zugriffs des Video Interface Chip auf den Charactergenerator, der sich (wie in Bild 2 zu sehen) direkt an den Basic-Speicher anschließt, ist dies nämlich nicht möglich. Dafür kann man diesen Bereich für Basic nutzen, denn der Basic-Interpreter läßt sich dementsprechend ändern. Daher meldet sich der VC 20 nach dem Starten des beigefügten Maschinenprogramms mit 32255 freien Bytes.

Schließlich ist auch der I/O-Bereich ($9000-$9FFF), der die VIC-Register, den Farbspeicher und die Ein-/Ausgaberegister enthält mit RAM-Speicher überlagerbar. Da dieser Speicherabschnitt bei meinem Testmodell nicht funktionierte, konnte ich nicht mit ihm arbeiten. Nach Auskunft des Herstellers kann man ihn sowieso nur mit einem bestimmten Programmiertrick nutzen, da der Prozessor während des Interrupts ständig auf diesen Bereich zugreift.

Da zu diesem Speicherabschnitt auch das Umschaltregister (Adresse $9FFF) gehört, kann es bei Bedarf über den mitgelieferten Schalter in eine ganz normale Speicherstelle umgewandelt werden.

An fertiger Software ist nach Angaben des Herstellers ein Forth-Compiler in Vorbereitung. In erster Linie ist man aber wohl auf seine eigenen Programmierkünste angewiesen. Prinzipiell kann man auch daran denken, Programme von C 64 auf den so erweiterten VC 20 umzuschreiben (wobei unterschiedliche Grafik- und Tonerzeugungsmethoden beachtet werden müssen).

Fazit: Will man diese Speichererweiterung voll ausnutzen, so benötigt man fundierte Maschinensprachekenntnisse. Unter diesem Gesichtspunkt gesehen bietet sie für Otto Normalbenutzer kaum Vorteile gegenüber den herkömmlichen 64 KByte-Karten. Wer jedoch das erforderliche Wissen besitzt, dem eröffnen sich mit dieser Karte völlig neue Aspekte der Programmierung, denn man hat — wie beim C 64 — die Organisation des ganzen Systems in der Hand.

(Christoph Sauer/ev)

Vorteile:
- Epansionsport bleibt frei
- vergrößerter Basic-Speicher
- Betriebssoftware ist änderbar
- Speicher-ICs sind gesockelt
- Alle Ausbaustufen des VC 20 können simuliert werden

Nachteile:
- Polungsrichtung für die CPU nicht angegeben
- nur für Maschinensprachenprofis voll nutzbar

Preis: 295 Mark
Roßmöller GmbH, Finkenweg 1, 5309 Meckenheim

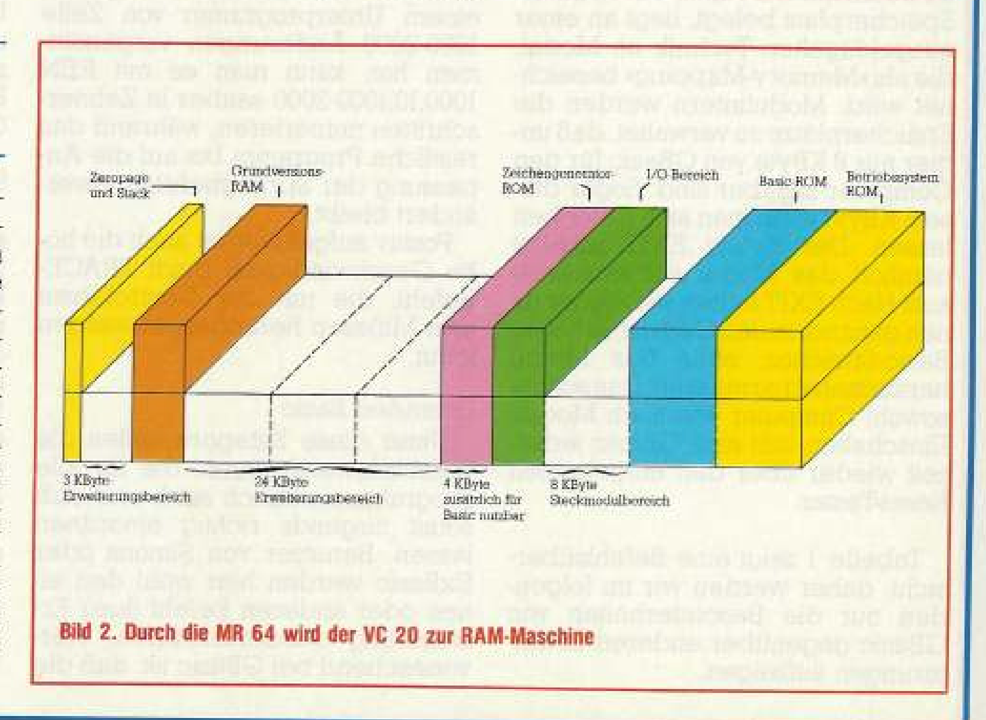
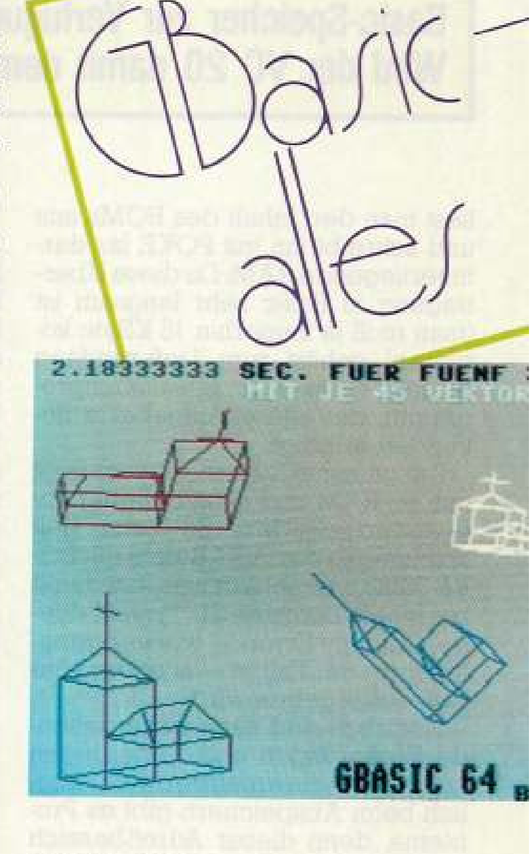


Bild 2. Durch die MR 64 wird der VC 20 zur RAM-Maschine

GBasic 64 ist eine relativ neue Basic-Erweiterung für den C 64. Sie ist wohl die leistungsfähigste, die es zur Zeit auf dem Markt gibt. GBasic 64 unterstützt sowohl Grafik als auch Sound. Verfechter der strukturierten Programmierung kommen ebenso zu ihrem Recht wie Anwender, die großen Wert auf Programmierhilfen legen. Dabei wird nicht nur Basic, sondern auch Assembler unterstützt.



GBasic ist noch relativ unbekannt, wahrscheinlich auch deswegen, weil es nur in einer Modul-Version vorliegt, die eine spezielle Hardware beinhaltet, und es somit praktisch kopierunfähig macht.

Das Modul macht einen sauberen soliden Eindruck. Goldkontakte, Standfüße, sowie einen eingebauten Reset-Taster sollte eigentlich jedes Modul haben. Mitgeliefert werden noch ein Handbuch, auf das ich noch zu sprechen kommen werde, sowie eine Diskette oder Kassette mit Demo-Programmen und Utilities. Nach dem Einstecken des Moduls in den Modulschacht und dem Einschalten des C 64 meldet sich dann auch sofort GBasic. Aus der Meldung wird ersichtlich, daß Sie nun 8 KByte weniger Speicherplatz für Basic-Programme zur Verfügung haben. Das Modul selbst hat allerdings 16 KByte ROM-Speicher. Daß trotzdem nur die Hälfte davon Basic-Speicherplatz belegt, liegt an einer ausgeklügelten Technik im Modul, die als »Memory-Mapping« bezeichnet wird. Modulintern werden die Speicherplätze so verwaltet, daß immer nur 8 KByte von GBasic für den Computer sichtbar sind. Sogar diese 8 KByte kann man sich freigeben lassen. Der Befehl EXIT schaltet nämlich das Modul softwareseitig aus! Nach EXIT haben sie wieder einen ganz normalen C 64 mit 38 KByte Basic-Speicher, ohne das Modul herausziehen zu müssen. Das schont sowohl Computer wie auch Modul. Einschalten läßt sich GBasic jederzeit wieder über den eingebauten Reset-Taster.

Tabelle 1 zeigt eine Befehlsübersicht, daher werden wir im folgenden nur die Besonderheiten von GBasic gegenüber anderen Erweiterungen aufzeigen.

Toolkit

GBasic besitzt einige Funktionen, die beim Programmieren sehr nützlich sind: Programmlistings können sowohl nach unten als auch nach oben gescrollt werden. Somit ist ein komfortables Editieren von Programmen möglich. Zwei nützliche und oft hilfreiche Befehle des Toolkits sind FIND- und der REN(umber).

FIND listet alle Basic-Zeilen, die eine bestimmte Zeichen- oder Befehlsfolge enthalten. Allerdings ist es nicht möglich, die Suche auf bestimmte Programmteile zu begrenzen. Bei häufigem Auftreten des gesuchten Begriffs leidet die Übersichtlichkeit etwas.

Dagegen ist der REN-Befehl komfortabel. Es werden beim Umnumerieren nicht nur alle GOTO, GOSUB, RESTORE-Befehle an die neuen Zeilennummern angepaßt, es kann auch bereichsweise umnumeriert werden. Wenn man zum Beispiel in einem Unterprogramm von Zeile 1000-2000 Änderungen vorgenommen hat, kann man es mit REN 1000,10,1000-2000 sauber in Zehnerschritten numerieren, während das restliche Programm bis auf die Anpassung der Sprungbefehle unverändert bleibt.

Positiv aufgefallen ist auch die hohe Geschwindigkeit beim TRACE-Befehl, die mit der Control-Taste zum Mitlesen herabgesetzt werden kann.

Extended Basic

Unter diese Kategorie fallen die Befehlserweiterungen, die für alle Programme nützlich sind, und sich somit nirgends richtig einordnen lassen. Benutzer von Simons oder ExBasic werden hier wohl den einen oder anderen Befehl ihrer Erweiterung wiedererkennen. Hervorstechend bei GBasic ist, daß die Befehle GOTO, GOSUB wie auch RESTORE nun auch mit Labels möglich sind. Diese Labels werden vor eine anzuspringende Zeile gesetzt. Schön wäre es noch gewesen, wenn man, wie in Simons Basic, globale und lokale Variablen zur Verfügung hätte.

Befehle wie IF..THEN..ELSE..,REPEAT..UNTIL, PRINT AT, PRINT USING, das Abfangen von ERRORs und so weiter, gehören schon fast zur »Standardausrüstung« von Basic-Erweiterungen und sind auch in GBasic enthalten.

Stringbefehle und Funktionen

Die Stringbefehle, die es in GBasic gibt, gleichen in ihrer Wirkung denen in Simons Basic. Man braucht keinen Haufen Unterprogramme, um Strings zu ersetzen, zum Einfügen und Suchen von Textteilen innerhalb eines Strings und so weiter. Neben dem EXOR-Befehl, der durch eine unübliche und unhandliche Syntax auffällt, wäre der BIT-Befehl zu erwähnen. Wenn man bestimmte Bits, zum Beispiel am Joystickport, auf gesetzt testen will, ist das mit einem Befehl möglich. BIT ist eine logische Funktion und kann somit direkt in IF-Abfragen eingesetzt werden.



Bild 1. Das alles kann der PRINT-Befehl

Bild 2. 3-dimensionale Figuren lassen sich mit einfachen Befehlen drehen

Als sehr flexibel und nützlich erweist sich der FUNCTION-Befehl. Mit ihm kann der Inhalt eines Strings berechnet werden. So kann man mit einem INPUT-Befehl Formeln eingeben, die FUNCTION auswertet. Im normalen Basic ist das nur mit großen Schwierigkeiten möglich. Zum Beispiel ergibt PRINT FUNCTION "12/4+3" die Ausgabe 6. FUNCTION arbeitet mit allen Basic-Funktionen und auch Variablen im Funktionsstring. Erleichtert wird die Programmierarbeit auch dadurch, daß jederzeit Hexadezimal- und Binär-Zahlen verwendet werden dürfen.

Peripherie

GBasic unterstützt nicht nur Diskettenoperationen, sondern auch Joystick, Lightpen und Paddle. Sehr interessant sind dabei DEV und BLOAD. DEV stellt die Standardgeräte-Adresse um, DEV8 zum Beispiel auf die Floppy. Jedes LOAD, SAVE und VERIFY bezieht sich jetzt, so lange nicht ausdrücklich anders angegeben, auf die Floppy-Disk. Sogar die Tastenkombination SHIFT-RUN/STOP lädt das erste Programm anstelle von der Datasette von der Floppy und startet es automatisch.

Mit BLOAD können Maschinenprogramme, Grafikbilder oder Spritedaten, kurz alles was nicht Basic ist, in den C 64 geladen werden. Wenn Sie sagen, »Das kann ich mit LOAD,X,1 auch«, dann werden Sie merken, daß im Gegensatz dazu BLOAD keinerlei Pointer verändert! Ein BLOAD-Befehl in einem Programm führt nicht zu einem Neustart nach erfolgtem Ladevorgang, sondern das Programm läuft ganz normal weiter. Auch der häßliche OUT OF MEMORY ERROR nach dem Nachladen von Maschinenprogrammen in den $C-Bereich tritt nicht mehr auf.

Noch ein paar Worte zum HCOPY-Befehl. Mit ihm kann hochauflösende Grafik ausgedruckt werden. HCOPY arbeitet mit jedem (!) grafikfähigen Acht-Nadel-Drucker zusammen. Diese Flexibilität mußte damit erkauft werden, daß Sie HCOPY erst mit einigen POKEs an ihren Drucker anpassen müssen. Eine Hardcopy der Textseite ist mit HCOPY nicht möglich.

Grafik

Kommen wir nun zu dem Teil von GBasic, der die meisten von Ihnen interessieren wird: die Grafik. GBasic verfügt über drei Grafikseiten, von denen zwei allerdings Basic-Speicherplatz belegen. Eine vierte kann noch als Zwischenspeicher verwendet werden. Bei ihr ist allerdings keine Farbsetzung möglich, da der Farbspeicher hier im GBasic-Modul selbst liegen müßte.

Zwischen diesen drei Seiten kann beliebig hin und her geschaltet werden, man kann sogar eine Seite bearbeiten, während eine andere oder die Textseite angezeigt wird. Es ist aber nicht möglich eine Multicolorseite anzeigen und eine »normale« Seite zu bearbeiten. Die Seiten können auch beliebig gemischt oder übereinander kopiert werden. Alle üblichen Zeichenbefehle sind implementiert: Punkte, Linien, Rechtecke und Blöcke können in sehr hoher Geschwindigkeit gesetzt, gelöscht oder invertiert werden.

Auch der CIRCLE-Befehl ist sehr vielseitig. Nicht nur Kreise, sondern auch Ellipsen, Vielecke, Kreisbögen und so weiter können gezeichnet werden. Dies allerdings in einer fast sensationellen Geschwindigkeit, vor allem, wenn man die Voreinstellung der Schrittweite ändert und erhöht. Die Kreise werden dadurch eckiger, aber sehr viel schneller gezeichnet. Hier hilft Experimentieren, bis man den optimalen Wert herausgefunden hat.

Ungewöhnlich, aber sehr nützlich ist der VECTOR-Befehl. Mit ihm kann man Linien zeichnen, die enden, sobald sie auf einen anderen gesetzten Punkt beziehungsweise Kreis, Linie oder sonstiges treffen. Schraffuren sind damit ein Kinderspiel. Der letzte gezeichnete Punkt wird in zwei Variablen gespeichert, so daß Sie volle Kontrolle über den Zeichenvorgang haben.

Nun aber zum gelungensten Befehl in GBasic. Gemeint ist PRINT. Ja, Sie haben richtig gelesen, PRINT. Sollten Sie jetzt den Kopf zweifelnd schütteln, so muß ich Ihnen sagen, daß ich in keiner Basic-Erweiterung einen solchen Komfort gefunden habe. Vergessen Sie TEXT (zum Beispiel aus Simons Basic) und seine Kollegen, mit denen Sie Texte in Grafiken eingebracht haben. Warum nehmen Sie nicht einfach PRINT? In GBasic funktioniert das, und zwar so gut, daß PRINT"shift-clr/home" tatsächlich die Grafikseite löscht. Alle Cursorbewegungen, Farbcodes, Groß/Klein-Umschaltungen und Revers-Schrift funktionieren wie auf der normalen Textseite, mit zwei kleinen Einschränkungen. Erstens ist das Ganze verständlicherweise etwas langsamer, und zweitens ist kein Scrolling nach oben möglich, wenn in die letzte Zeile geschrieben wird.

Doch damit nicht genug. Sie können mit PSN den Cursor fein, das

heißt einzelpunktweise, positionieren. Nach PSN 0,1 steht der Cursor (bitte bedenken Sie, daß er während der Programmausführung nicht zu sehen ist) um eine Einzelpunktzeile tiefer als nach PRINT"clr/home" und am linken Rand. Mit PSN ist zum Beispiel Schrägschrift oder ein Funktionenplotter mit Print-Befehlen möglich. Eine weitere Manipulationsmöglichkeit ist mit SIZE gegeben. Sie können damit die Zeichengröße in X- und Y-Richtung beliebig vervielfachen, bis aufs 38x24 = 912-fache. Das wäre dann allerdings bildschirmfüllend.

Schließlich kann noch der Zeichenabstand manipuliert werden, dies allerdings nur mit POKE-Befehlen. So läßt sich eine Engschrift, oder gar Proportionalschrift realisieren. Alle diese Möglichkeiten des PRINT-Befehls habe ich in Bild 1 ausgeschöpft. Aber das ist immer noch nicht alles. Mit der GBasic-Demo-Diskette/Kassette erhalten Sie noch eine Befehlserweiterung zu GBasic, die auf einfache Art und Weise 3D-Grafik ermöglicht. Dazu müssen Sie ersteinmal Shapes erstellen und im Speicher ablegen. Ein Shape ist, salopp gesagt, ein dreidimensionaler Linienzug beliebiger Länge und Komplexität. Haben Sie solch ein Shape erstmal definiert, können sie es mit drei zusätzlichen Befehlen beliebig um die Raumachsen drehen beziehungsweise vergrößern, um es danach im hochauflösenden Grafik-Modus zeichnen zu lassen. Gezeichnet wird aus Geschwindigkeitsgründen parallelperspektivisch im Drahtgittermodell, das heißt, eigentlich unsichtbare Linien werden mitgezeichnet. Ein Beispiel sehen Sie in Bild 2.

Sprites

Ähnlich reichhaltig wie der Grafikbefehlssatz ist der für die Sprites. Das fängt schon mit einem ebenfalls auf der Demo-Diskette/Kassette befindlichen Spriteeditor an, dessen Spritedaten Sie in GBasic direkt weiterverarbeiten können. Alle Spriteparameter sind über Befehle erreichbar. Es ist auch möglich, Sprites interruptgesteuert und unabhängig vom Basic-Programm bewegen zu lassen. Sprite-Sprite- und Sprite-Hintergrund-Kollisionen können auch jederzeit in GBasic abgefragt werden. Dies ist allerdings auch interruptgesteuert möglich, bei einer Kollision wird dann in ein Basic-Unterprogramm gesprungen. Hier kann sogar nur auf bestimmte Kollisionen, wie Sprite drei mit Sprite sieben oder ähnliches, geprüft werden.

Befehl	Funktion
Toolkit	
AUTO	—Automatische Zeilennummern
REN	—Zeilenumnumerierung
DEL	—Löscht Programmbereiche
FIND	—Suche nach Ausdruck im Programm
DUMP	—Ausgabe aller Variablen
TRACE	—aktuelle Prg.Zeile anzeigen
OLD	—Rückgängigmachen von NEW
KEY	—Funktionstasten belegen
DISP	—Funktionstastenbelegung anzeigen
EXIT	—Schaltet GBasic-Modul aus
Extended Basic	
PAUSE	—wartet einen angegebenen Zeitraum
ELSE—	Für IF.THEN..ELSE..
LASTIF	—Letzten IF-Befehl weiterführen
REPEAT.UN-TIL	—Schleife mit Schlußbedingung ähnlich PASCAL
POP	—vorzeitiger Ausstieg aus Schleife oder Unterprogramm
LBL	—legt Label für Sprunganweisung/RESTORE fest (GOTO/GOSUB/RESTORE "TEST")
CASE ERR GOTO	—»Abfangen« von Fehlermeldungen
OFF	—schaltet CASE ERR GOTO ab
RESTORE	—funktioniert nun auch mit Zeilennummern
PRINT USING	—Formatierte Zahlenausgabe
VTAB	—Cursor in Bildschirmzeile setzen
SWAP	—Vertauscht zwei Variableninhalte
Strings	
INSTR	—Suchen eines Strings in einem anderen
REPL$	—Teilweises Überschreiben/Ersetzen eines Strings
INST$	—Einfügen eines Strings in einen anderen
MUL$	—Vervielfachen eines Strings
Eingabe	
FETCH$	—Verbessertes GET
INLINE$	—Verbesserter INPUT
BUT	—Funktionstasten abfragen
Funktionen	
FUNCTION	—Berechnen eines Strings
FRAC	—Vorkommastellen einer Zahl abschneiden
MOD	—Modulo-Funktion, Rest einer Division
EXOR	—Verknüpft zwei Zahlen logisch exclusiv-oder
BIT	—Testet auf bestimmte Bits
$	—Kennzeichnet Hexadezimalzahl
%	—Kennzeichnet Binärzahl
HEX$	—wandelt Dezimal in Hexadezimal
BIN$	—wandelt Dezimal in Binär
Speicher	
DOKE	—POKE für Adreßwerte im HI/LO-Format
DEEK	—PEEK für Adreßwerte im HI/LO-Format
LOMEM	—legt Untergrenze des BASIC-Speichers fest
HIMEM	—legt Obergrenze des BASIC-Speichers fest
Peripherie	
DEV	—stellt die Standardgeräteadresse um
DIR	—zeigt Disketten-Directory an
DISK	—sendet ein Kommando an die Floppy-Disk
ERR	—liest den Kommandokanal aus
MERGE	—hängt ein Programm an ein anderes an
BLOAD	—LOAD ohne Veränderung von Variablen
BSAVE	—Abspeichern eines Speicherblockes
HCOPY	—Hardcopy der hochauflösenden Grafik
JOY	—Abfrage eines der beiden Joysticks
PEN	—Abfrage eines Lightpen
PDL	—Abfrage eines der vier Paddles
Grafik	
HGR	—schaltet Grafik ein (normal/Multicolor)
TEXT	—schaltet auf Textbildschirm zurück
COLORG	—Farben für die hochauflösende Grafik
COLORT	—Farben für die Textseite
COL	—Zeichenfarbe ändern.
INK	—Zeichenfarbe wählen.
MODE	—Betriebsart in hochauflösender Grafik zeichnen, löschen, invertieren, zählen
SCREEN	—Auswahl des verwendeten Grafikschirms
ADD	—überlagert oder kopiert Grafikschirme
PLOT	—zeichnet Punkt
LINE	—zeichnet Linie
VECTOR	—zeichnet Linie bis zum nächsten Punkt
CIRCLE	—zeichnet Kreise, Ellipsen, Kreisbögen etc.
BOX	—zeichnet Rechteck
BLOCK	—zeichnet Balken
FILL	—Ausfüllen einer umrandeten Fläche
PRINT	—zeichnet Text/Grafikzeichen in der HGR
SIZE	—verändert die Zeichengröße für PRINT
PSN	—Feinpositionierung der Texte in HGR
MEM	—Umschaltung Zeichensätze original/eigener
Sprites	
SEDIT	—ruft den Spriteeditor auf
SPRITE	—legt das Aussehen eines Sprites fest
SCOL	—Spritefarben festlegen
SPOS	—setzt ein Sprite an eine bestimmte Position
SMOV	—Interruptgesteuerte Bewegung von Sprites
SPRX/SPRY	—Position eines Sprite feststellen
CHECK	—Test auf Kollision Sprite-Sprite/Hintergrund
COND.GOSUB	—Interruptgesteuerte, dauernde Überwachung von Kollisionen, bei Kollision GOSUB
RTN	—Rückkehr aus COND.GOSUB
SOFF	—Sprite wieder abschalten
Musik	
VOL	—Lautstärke einstellen
ENVELOPE	—Hüllkurve einer Stimme einstellen
WAVE	—Wellenform, Ring-, und Sync. einstellen
WIDTH	—Pulsbreite bei Rechteckschwingung
SEFILT	—Filter einstellen
FILTER	—Stimmen auf Filter leiten
SND	—Umrechnung Notenname-Frequenz
TUNE	—Ton spielen
PLAY	—Musikstück interruptgesteuert spielen
VOFF	—Stimme abschalten
Monitor	
TIM	—ruft den eingebauten Monitor auf. —Möglichkeiten Load, Save, Verify Assemble, Disassemble Hex-, ASCII-, Binary-, Dezimal-Dump & Eingabe Verschieben mit/ohne Adressenangleichung Suchen beliebiger Zeichenfolge mit Joker Trace für Maschinenprogramme Berechnungen und Variable im Monitor

◄ **Tabelle 1. Alle GBasic-Befehle auf einen Blick**

Musik

Der letzte Bereich von neuen Basic-Befehlen betrifft den Sound-Chip. Auch hier zeigt sich GBasic als praktisch. Es kann auf jeden Parameter des Sound-Chips zugegriffen werden, ohne zu POKEn. Sogar der Filter ist in GBasic voll steuerbar. Aber damit nicht genug. Für Soundeffekte reichen die normalen Soundbefehle voll aus, aber sobald man längere Musikstücke spielen möchte, hört der Spaß auf. Deswegen wurde in GBasic noch eine Programmiersprache, MUSIC, integriert. MUSIC hat 14 Befehle, die das Programmieren von Musikstücken zum Kinderspiel machen. Diese sind nicht in Tabelle 1 aufgeführt! Ein in MUSIC programmiertes Lied wird irgendwo im Speicher abgelegt. Es kann dann mit dem PLAY-Befehl gestartet werden und läuft interruptgesteuert bis zum Ende oder endlos, je nach Programmierung. Während die Musik spielt, kann ein Basic-Programm unbehindert nebenher laufen. Die Eingabe von MUSIC-Programmen würde allerdings wieder zur POKEerei werden, wenn Omikron nicht auch noch einen MUSIC-Editor mitliefern würde. Der ist ebenfalls auf der Demo-Diskette/Kassette.

Monitor

Und um das Maß und die 16 KByte Speicher voll zu bekommen, ist in GBasic auch noch ein Maschinensprachemonitor implementiert. Er beherrscht so ziemlich alles, was man von einem komfortablen Monitor erwartet, den man sich einzeln, also ohne Basic-Erweiterung zulegen würde. Einziger Kritikpunkt ist die ungewöhnliche Bedienung über die Funktionstasten. Deswegen wurde auch hier keine genaue Befehlsauflistung gegeben. Alle Fähigkeiten des Monitors finden sich aber in Tabelle 1. Dieser Monitor ist wohl für jeden normalen Anwendungsfall ausreichend.

Dokumentation

Das mitgelieferte Handbuch muß hier ausdrücklich gelobt werden. Es werden nicht nur alle Befehle gut

und genau erklärt, auch sind sehr viele Hintergrundinformationen enthalten, wie zum Beispiel eine Speicherbelegung von GBasic. Es wird auf die mitgelieferten Programme genauso stark eingegangen wie auf GBasic selber. Eine Befehlsübersicht und einige wichtige Tabellen runden das Gesamtbild ab. Ich vermisse nur noch eine Zeropagebelegung von GBasic. Warum? Nun, GBasic läßt sich vom Benutzer beliebig erweitern. Wenn Sie einen neuen Befehl in GBasic einbauen wollen, müssen Sie ihm nur ein "!" voranstellen. Findet GBasic ein "!", springt es nach $C000, wo Sie dann Ihre Befehlsauswertung vornehmen können. Ein Beispiel für eine solche Erweiterung ist die nachladbare 3D-Grafik.

Wie Sie sehen, bietet GBasic eine ganze Menge, und zwar nicht nur von jedem etwas, sondern jeder seiner Befehlsbereiche ist in sich geschlossen und vollständig. Beim Test habe ich nur einen Fehler entdecken können. Wenn Sie mit DISK ein Kommando an die Diskette senden wollen, diese aber nicht eingeschaltet ist, hängt sich GBasic auf, und läßt sich nur noch mit RUN/STOP-RESTORE wiederbeleben. Ähnliches passiert bei DIR. Es wird kein DEVICE NOT PRESENT ERROR ausgegeben, sondern einfach nur READY. Diese beiden Fehler sollen aber in einer neuen Version, die gerade bei Omikron fertiggestellt wird, beseitigt werden. Zusätzlich will man dort ein schnelleres Laden von Diskette, ähnlich Hypra-Load (64'er, 10/84), einbauen. Trotzdem soll volle Kompatibilität zu der alten GBasic-Version bestehen bleiben.

Abschließend kann ich sagen, daß mich GBasic überzeugt hat. Ausschlaggebend waren die enorme Befehlsvielfalt, wie auch die hohe Geschwindigkeit der meisten Funktionen. Ich hoffe, daß diese Erweiterung etwas bekannter, und vielleicht, ähnlich wie für Simons Basic, fertige Software dafür geschrieben wird. Auch die Erweiterbarkeit von GBasic kann ein Anreiz für andere Programmierer sein. Insgesamt ein Paket, das seine 259 Mark wert ist.

(Boris Schneider/gk)

Bezugsquelle:
Omikron Software, Erlachstr.15, 7534 Birkenfeld 2

ASSEMBLER?

Mein Weg zur Maschinensprache begann mit den vielen, vielen DATAs, die sicherlich jeder von uns einmal eingegeben hat, ohne überhaupt zu wissen, welche Bedeutung sie hatten. Diese Unmenge an Zahlen verschwand dann in der Tiefe des Rechners, wurde nicht mehr gesehen und vollbrachte wahre Wunderdinge.

Also entschloß ich mich eines Tages, einen in Basic geschriebenen Maschinensprach-Monitor abzutippen und damit zu arbeiten. Plötzlich erschienen gar seltsame Zahlen und Buchstaben auf dem Bildschirm:

4000 A0 00 B9 00 41 F0 06 20

So ungefähr muß es in grauer Computer-Urzeit auf meinem Bildschirm ausgesehen haben.

Nun, Sie müssen zugeben, daß diese »Hieroglyphen« sicherlich nicht gerade aufschlußreich erscheinen. Mit einer Befehlsliste für den 6502 ging ich nun daran, das »Zahlenrätsel« zu lösen. Zum ersten Mal konnte ich meinem Computer direkt in das Innerste — also geradezu in die Eingeweide — schauen. Allerdings war dieses Verfahren der Übersetzung auf die Dauer sehr eintönig und sehr, sehr zeitraubend. Also wie geschaffen für einen Computer.

Sehen wir uns noch einmal das obengenannte Beispiel genauer an:

Bei der ersten Angabe (4000) handelt es sich um die Adresse, in der der folgende Wert (A0) steht. Dies ist ein Befehl oder »Operationscode«, der eine Operation in der CPU des Rechners bewirkt: Er lädt das Y-Register (im Englischen abgekürzt — LOAD Y) mit der folgenden Zahl (00).

Da »unser« C 64 einen Befehlsvorrat von 50 Befehlen kennt, können Sie sich vorstellen, daß es sehr anstrengend ist, sich diese Anzahl an Befehlen in Form von Hex-Zahlen zu merken. Hinzu kommt noch, daß durch die verschiedenen Adressierungsarten die Gesamtzahl auf 150 Befehle ansteigt, so daß es selbst für einen Geistesakrobaten schwierig wird. Deshalb kamen findige Tüftler auf die Idee, Mnemonics (= Gedächtnishilfen) zu »erfinden«, die in 3 Buchstaben das Entscheidende des Befehls ausdrücken:

Aus A0 (LOAD Y) wird als Mnemonic LDY. Aus B9 (LOAD AKKU) wird LDA!

Wenn wir uns jetzt diese Erleichterung zu Nutze machen, können wir unser Maschinen-Programm auch folgendermaßen schreiben:

```
LDY  #00
LDA  4100,Y
BEQ  400D
JSR  FFD2
```

Nun fällt uns auch die Übersetzung des Programmes viel leichter:

1. lade das Y-Register mit dem Wert 00,
2. lade den Akku mit dem Wert, der in Adresse 4100+Y steht,
3. springe, wenn dies eine Null war, nach 400D,
4. springe ansonsten in das Unterprogramm (JSR = JUMP SUBROUTINE), das bei FFD2 beginnt.

Sie werden bemerkt haben, daß wir uns immer mehr von der »reinen« Maschinensprache entfernen, da wir bemüht sind, das Programmieren für uns Menschen verständlicher zu machen.

Wir benötigen also ein Programm, welches in der Lage ist, einen so eingegebenen Text zu übersetzen. Solch ein Programm, das einen in Mnemonics (zum Beispiel LDA..., STA...) vorliegenden Text in Maschinencode übersetzt, nennt man einen **Assembler.**

Entsprechend wird ein umgekehrt arbeitendes Programm als **Disassembler** bezeichnet. Dieser hat also den Vorteil, daß er unsere Programmzeile nicht in reinen Hexadezimalzahlen (A0, B9, ...), sondern in Mnemonics (LDY #00, LDA 4100, Y ...) ausdruckt.

Leider heißt auch die Programmiersprache, bei der Mnemonics benutzt werden, **Assembler(-Sprache),** so daß hierbei leicht Verwechslungen auftreten.

Neben dieser reinen Übersetzungstätigkeit besitzen fast alle Assembler(-Programme) aber noch weitere Möglichkeiten, die Arbeit zu erleichtern. Alle guten Assembler erlauben den Einsatz von **Labels.** Dabei handelt es sich um freigewählte Namen, die anstelle der absoluten Werte gesetzt werden. In unserem Beispiel könnten wir die absoluten Adressen hinter LDA und JSR durch einen beliebigen Namen ersetzen. Unser Beispiel könnte also etwa so aussehen:

```
        LDY  #
LOOP    LDA  TEXT,Y
        BEQ  ENDE
        JSR  AUSGEBEN
        INY
        BNE  LOOP
ENDE    ...  ...
```

Ein Vorteil dieser Prozedur liegt vor allem darin, daß diese Namen

ASSEMBLER!

Wer die Fähigkeiten seines Computers voll und ganz ausnutzen möchte, kommt um den Einsatz der Maschinensprache nicht herum. Schließlich ist sie die einzige, die der Computer wirklich versteht. Allerdings hat sie den entscheidenden Nachteil, daß der Mensch mit einer Anhäufung von Nullen und Einsen kaum etwas anfangen kann. Kluge Köpfe haben deshalb die Assembler-Sprache erfunden, einen Kompromiß, mit dem beide, Mensch und Computer, zufrieden sein können.

schon beim Lesen des Quelltextes erkennen lassen, welche Funktion zum Beispiel das Unterprogramm »Ausgeben« hat. Anders als in Basic, wo ein GOSUB 12600 nichts darüber aussagt, was dort geschehen soll. Ein weiterer Vorteil ist, daß wir uns beim Schreiben des Textes (auch als **Quelltext,** Quellcode beziehungsweise Sourcecode bezeichnet) nicht von vornherein über sämtliche Sprungziele im Klaren sein müssen. Wo zum Beispiel der Text steht, ist erst einmal völlig gleichgültig. Wir könnten auch noch später einige Befehle zwischen LOOP und ENDE einfügen, kurz, der Quelltext ist ohne großen Aufwand beliebig veränderbar. Wären statt dessen absolute Adressen verwendet worden, müßten diese bei jeder Änderung ebenfalls angepaßt werden. Bei längeren Programmen ist das nahezu unmöglich. Die Rechnerei übernimmt nun der Assembler: Dazu arbeitet er den Quelltext in — normalerweise — zwei Schritten **(Pass)** durch. Im ersten Pass werden alle Sprungziele etc. berechnet und in einer **Symboltabelle** abgelegt, im zweiten Pass wird das Maschinenprogramm (**Objectcode,** Maschinencode) im Speicher abgelegt.

Wenn in einem Programm verschiedene Teile mehrfach auftauchen, ist es sicherlich nicht sinnvoll, diese jedesmal neu eingeben zu müssen. Deshalb bieten verschiedene Assembler die Möglichkeit, solche Teile als **»Makro«** zu definieren. Anstelle der gesamten Befehlsfolge genügt es, nur den vorher definierten Makronamen einzusetzen. Unser Beispielprogramm ließe sich mit dem Makro-Namen »Textaus« versehen und dann an jeder Stelle im Programm aufrufen, die einen Text ausgeben soll.

Als weitere Bequemlichkeit nehmen die meisten Assembler dem Programmierer das Umrechnen der verschiedenen Zahlensysteme ab. Sie verarbeiten Binärzahlen ebenso wie Dezimal- oder Hex-Zahlen. Häufig kann man auch Texte als Buchstabenfolge und nicht in ASCII-Codes eingeben. Darüber hinaus sind beim Operanden einfache Rechen- beziehungsweise logische Operationen erlaubt, zum Beispiel:

```
LDA   #53280 AND 255
oder
STA   TEXT+7,X
```

Um dem Assembler Anweisungen beim Übersetzen zu geben, werden **Pseudo-Opcodes** eingesetzt. So bedeutet zum Beispiel:

```
.BA $C000
```

Beginne die Assemblierung bei $C000. (Bisweilen wird hierfür auch der Pseudo-Opcode »ORG« benutzt. ».BY« oder ».WORD« signalisiert dem Assembler, daß die folgenden Zeichen nicht als Befehl, sondern als Byte-Folge im Speicher abgelegt werden sollen. Ebenso gibt es Pseudo-Opcodes, die die Ausgabe eines Assemblerlistings auf dem Drucker steuern. In der Anzahl und den Möglichkeiten der Pseudo-Opcodes unterscheiden sich die verschiedenen Assembler stark voneinander.

Zur Eingabe des Quelltextes wird ein **Editor** benötigt. Im einfachsten Falle ist dies der von Basic gewohnte Editor. Gute Assembler bieten hier jedoch weitaus mehr. Befehle zum Suchen (FIND) und Ändern (EDIT) bestimmter Befehlsfolgen gehören eigentlich zum Standard. Befehle zum Blättern im Quelltext, zum Einfügen, Kopieren und Verschieben von Textteilen, wie man sie aus Textverarbeitungssystemen kennt, bieten ein Höchstmaß an Komfort. Auch hierin unterscheiden sich die Assembler ganz erheblich. Der Editor erlaubt meistens auch, Teile des Programms separat abzuspeichern und sich so eine Bibliothek von häufig benötigten Unterprogrammen anzulegen. Damit wären die wichtigsten Begriffe, die im Zusammenhang mit Assemblern auftauchen (zum Beispiel in den Testberichten in dieser Ausgabe) erläutert. Eine Zusammenfassung finden Sie in unserem »Kleinen Assemblerlexikon«.

(N. Mann/D. Weineck/gk)

Monitor
Programm zum Bearbeiten von Maschinenprogrammen. Enthält mindestens HEX-DUMP, DISASSEMBLER sowie LOAD- und SAVE-Befehle. Bessere Monitore bieten zum Beispiel die Möglichkeit, Programme unter Kontrolle der Register ablaufen zu lassen.

Hex-Dump
(Memorydump, Speicherdump). Auflisten eines Maschinenprogrammes in Hex-Zahlen.

Disassembler
listet ein Maschinenprogramm in Form von MNEMONICS auf.

Assembler
übersetzt einen in Assemblersprache geschriebenen QUELLTEXT in OBJECTCODE.

Mnemonic
Aus drei Buchstaben bestehende »Gedächtnishilfe«, Abkürzung eines Assemblerbefehls.

Label
(Marke), ein (sinnvoller) Name, der bei der Assemblierung in eine feste Zahl umgerechnet wird. Labels können Sprungziele, aber auch Operanden sein.

Makro
Häufig auftretende Programmteile werden unter einem Namen zusammengefaßt. Der Quelltext enthält nur den Namen, beim Assemblieren wird die Befehlsfolge eingesetzt.

Quelltext
(Sourcecode) ist der Text, den man mit Hilfe des EDITORS in Assembler-Sprache schreibt. Er wird später vom Assembler übersetzt.

Objectcode
(Maschinencode) ist das vom ASSEMBLER erzeugte, fertige Maschinenprogramm.

Pseudo-Opcode
enthält Anweisungen an den Assembler.

Symboltabelle
wird vom Assembler im ersten Pass erstellt. Sie enthält die absoluten Adressen aller LABELs.

Kleines Assemblerlexikon

Assembler im Test

Teil 1

Mit dem AS-64 von Roßmöller erhält man nicht nur einen Assembler, sondern ein ganzes Assemblerpaket: Eine EPROM-Karte enthält in 16 KByte den Editor, den Assembler und einen einfachen Monitor. Dazu gibt es eine Diskette mit weiteren Hilfsprogrammen: Einen Re-Assembler, einen Monitor mit Disk-Monitor und einen Demo-Quelltext. Damit gehört AS-64 sowohl vom Umfang, allerdings auch vom Preis her zu den »besseren« Assemblersystemen für den C 64.

Fast ein Textsystem: Der Editor

Der Text wird so editiert wie man es von Textprogrammen her gewöhnt ist: Man schreibt einfach drauflos. Eine (abschaltbare) Tabulatorfunktion sorgt dafür, daß der Text schon beim Schreiben formatiert ausgegeben wird, das heißt Labels stehen am Anfang der Zeile, die Befehle beginnen in Spalte 12. Die zur Formatierung notwendigen Leerzeichen werden allerdings nicht mit abgespeichert, sondern nur bei der Ausgabe auf Bildschirm oder Drucker eingefügt. Dies spart natürlich enorm Speicherplatz. Änderungen erfolgen durch einfaches Überschreiben des alten Textes.

AS-64: auf Modul und Diskette

Ansonsten gibt es alles, was man von Textverarbeitungen her kennt: »Cursor up« und »Cursor down« ermöglichen zeilenweises, »Home« und F1 seitenweises »Blättern« vorwärts und rückwärts. Mit »SHIFT Home« gelangt man an den Anfang des Textes, mit F2 ans Ende. Mit F3 wird eine ganze Zeile eingefügt, mit F4 gelöscht. Die Tasten F7 und F8 erfüllen eine Spezialfunktion für Labels. Mit F7 werden sechs Leerzeichen eingefügt, mit F8 kann man von der Cursorposition bis zum linken Rand löschen. Damit lassen sich bequem zusätzliche Labels einbauen oder löschen. Wie der Assembler ist auch der Editor auf maximale Geschwindigkeit ausgelegt. Der Quelltext wird als Bildschirmcode im Speicher abgelegt, so entfällt eine dauernde Umrechnung in ASCII. Die damit erreichte Geschwindigkeit ist enorm. 30 KByte, das entspricht zirka 35 A4 Seiten Quelltext, kann man mit F1 in weniger als 10 (!) Sekunden durchblättern. Ich kenne kein Textsystem, mit dem das möglich wäre.

Natürlich hat dies seinen Preis: Schreibt man bei einer Änderung nämlich über ein Zeilenende hinaus, ohne vorher mit »Insert« die erforderliche Anzahl Leerzeichen einzufügen, springt der Editor in die nächste Zeile und überschreibt den dort stehenden Text. Eingewöhnung ist also nötig. Besser wäre hier eine Art »Insert-Modus«, mit dem man automatisch Text einfügen könnte.

Mit »CTRL-Z« werden Zeilen oder Textteile markiert, die anschließend verschoben, kopiert, gedruckt oder gelöscht werden können. Mit »CTRL-*« lassen sie sich auch auf Kassette oder Floppy abspeichern, nützlich, wenn man sich eine Bibliothek häufig benutzter Unterprogramme anlegen will.

Alle weiteren Editorkommandos entsprechen den Funktionen komfortabler Textsysteme. Neben einem »FIND«-Modus, der beliebige Zeichenketten sucht, gibt es eine »EDIT«-Funktion. Damit lassen sich ganze Worte durch andere ersetzen (zum Beispiel Label durch *Label, wenn man ein Label nachträglich in die Zeropage verlegen möchte.) Auch hier ist auf extreme Geschwindigkeit geachtet worden.

Besondere Erwähnung verdient auch der Umgang mit Peripherie-Geräten: Ein am User-Port angeschlossener Drucker wird automatisch erkannt und über eine eingebaute Centronics-Schnittstelle angesprochen. Commodore-Drucker werden wie üblich über die serielle Schnittstelle versorgt.

Als externe Speicher können Floppy oder Datasette angeschlossen werden. Beim Umgang mit der Floppy steht selbstverständlich eine Directory-Funktion zur Verfügung, ebenso können Kommandos an die Floppy ausgegeben werden. Eine Spezialität ist das »Backup«: Damit wird auf der Diskette eine Sicherheitskopie des letzten Textes erzeugt und gleichzeitig der neue gespeichert. So hat man immer die beiden (!) letzten Versionen zur Verfügung.

Der Assembler

Er kann erst einmal alles, was man von einem guten Assembler erwartet: Labels mit maximal elf Zeichen werden verarbeitet, Operand-Eingaben können dezimal, hexadezimal, binär und auch als ASCII-Zeichen erfolgen. Allerdings muß die Zeropage-Adressierung dem Assembler mit einem »*« vor der Adresse angezeigt werden, sonst wird sie als absolute Adressierung, also als 3-Byte-Befehl, assembliert.

Daneben gibt es eine Reihe von Pseudo-Opcodes, die Anweisungen an den Assembler enthalten. So kann man zum Beispiel »auf Probe« assemblieren, ohne den erzeugten Maschinencode (Objektcode) gleich in den Speicher zu schreiben. Man ist dadurch zumindest vor Syntax-Fehlern sicher und überschreibt nicht aus Versehen den Quelltext. Mit .LS und .LC können beliebige Teile des Programms während der Assemblierung auf Bildschirm oder Drucker aufgelistet werden. Ein besonders bequemes Kommando: Die .BY-Anweisung gestattet nicht nur die Eingabe von einzelnen Bytes — etwa einer Tabelle —, sondern vor allem auch die bequeme Eingabe von Texten. Diese werden einfach als Buchstabenfolgen abgelegt und beim Assemblieren in die zugehörigen ASCII-Werte umgerechnet. Damit entfällt das lästige Wälzen von ASCII-Tabellen.

Weitere Pseudo-Opcodes unterstützen dies: So ist es möglich, bei der Assemblierung bestimmte Teile des Quelltextes zu überspringen (bedingte Assemblierung). Man kann damit ein und dasselbe Quellprogramm für verschiedene Rechnertypen benutzen. Bei der Adreßeingabe sind nicht nur Labels, sondern auch noch umfangreiche Re-

Wer in Maschinensprache programmieren möchte, benötigt einen Assembler. Und da gerade aus dem C 64 mit Maschinensprache sehr viel herausgeholt werden kann, haben wir die interessantesten Assembler für Sie getestet. Sie lassen sich in zwei Preisgruppen einteilen, die unter und die ab 100 Mark. Im ersten Teil ist die obere Preisgruppe mit AS-64, MAE, T.EX.AS und ASSI/M an der Reihe.

chenoperationen gestattet. Es stehen die vier Grundrechenarten sowie logische Verknüpfungen (AND, OR, EOR) zur Verfügung. Damit läßt sich ein Quelltext sehr flexibel halten, nur wenige Adressen müssen fest vorgegeben werden, der Assembler selbst rechnet den Rest aus und paßt ihn automatisch an. Im Normalfall genügt dann die Änderung der Programmstartadresse (.BA) und das Programm läuft in einem anderen Speicherbereich. Alle bisher genannten Funktionen sind auch schon früher in guten Assemblersystemen vorhanden gewesen. Auch die Syntax der Befehle dürfte jedem, der mit einem MAE, ASTEX oder ähnlichen Assembler für den 6502 bereits gearbeitet hat, geläufig sein. Zwei Merkmale aber sind es, die AS-64 herausheben: Das eine und zweifellos wichtigste ist die fast unglaubliche Geschwindigkeit. Zwei Beispiele:

Einen 14 KByte langen Quelltext (Super Copy) assembliert AS-64 in fünf (!) Sekunden. Der bisher schnellste mir bekannte Assembler (MAE) braucht dafür immerhin 65 Sekunden. Aus einem 29 KByte langen Quelltext macht AS-64 in 11 Sekunden zirka 5 KByte Objektcode.

Erhöhter Zeitbedarf ergibt sich nur dann, wenn die Quelltexte länger als 30 KByte werden. Dann müssen nämlich Teile von Diskette nachgeladen werden. Auch diese Art der Assemblierung unterstützt AS-64: Wird ein Quelltext nicht mit ».EN«, sondern mit ».CT "name"« abgeschlossen, wird automatisch das Quelltextteil »name« von Diskette geholt und assembliert. Damit ist die Verarbeitung von beliebig langen Maschinenprogrammen möglich. Allerdings dürfte das nur in den seltensten Fällen nötig sein. Durch geschickte Speicherverwaltung sind bei AS-64 nämlich Quelltexte bis zu 30 KByte Länge möglich. Beim Assemblieren wird der Editor abgeschaltet, so daß im darunterliegenden RAM die Labeltabelle angelegt werden kann. Der erzeugte Objectcode kann mit dem Pseudo-Opcode ».MC« zum Beispiel nach $E000 unter das Betriebssystem gelegt und von dort später mit einer Spezialfunktion des Monitors wieder an die richtige Adresse verschoben werden. Dies ist das zweite große Plus von AS-64.

Fehlermeldungen bei der Assemblierung werden übrigens im Klartext und auf Deutsch ausgegeben (Sprung zu weit, Label doppelt etc.). Der Assembler stoppt dann die weitere Assemblierung. Allerdings erhält man keinen Hinweis darauf, an welcher Stelle im Quelltext der Fehler aufgetreten ist, vielmehr ist man auf die »FIND«-Funktion des Editors angewiesen. Angenehmer wäre es, wenn bei einem Fehler automatisch der Editor angesprungen und die fehlerhafte Zeile markiert würde.

Apropos Programmabsturz: Mir ist es noch nie gelungen, auch nur das kleinste Maschinenprogramm auf Anhieb so zu schreiben, daß es sich nicht beim ersten Probelauf »verabschiedet«. Bisher zog das eine recht umständliche Prozedur nach sich. RESET drücken, Assembler neu laden, Quelltext neu laden, neu probieren. Mit AS-64 sieht das anders aus. Da sich das Programm im EPROM befindet, kann man es auch bei noch so »geschickter« Programmierung nicht zerlegen. Nach RESET befinde ich mich wieder im Anfangs-Menü und kann einen »Softstart« durchführen. In den meisten Fällen ist dann der Quelltext sogar noch vorhanden, und es kann sofort weitergehen.

Der Monitor

Ein kleiner Monitor ist direkt ins System integriert. Seine Funktionen erlauben Disassemblieren und Verschieben des Objektcodes sowie das Laden und Abspeichern fertiger Programme. Braucht man mehr, zum Beispiel Such-Funktionen oder einen Trace-Modus, muß man den »großen« Monitor von Diskette laden. Ebenfalls auf der Diskette befindet sich ein Re-Assembler, mit dem man fertige Maschinenprogramme wieder in editierfähigen Quelltext zurückverwandeln kann. Auch Labels werden wieder gesetzt, aber natürlich nicht mit sinnvollen Namen versehen. Mit dem Re-Assembler kann der Anwender nach Belieben Maschinenprogramme seinen eigenen Wünschen anpassen, was bisher nur bei Basic-Programmen möglich erschien.

Makros sind aus Speicherplatzgründen nicht im EPROM untergebracht, sondern müssen ebenfalls bei Bedarf von Diskette geladen werden. Mir lag dieser Programmteil leider noch nicht vor, er soll aber bei Erscheinen dieser Ausgabe vorhanden sein. Außerdem kündigt der Hersteller an, daß eine erweiterte Version (mit 32 KByte EPROM) geplant ist, in der dann auch Makros und ein leistungsfähiger Monitor integriert sein werden. Der Besitzer von AS-64 kann später seine Version aufrüsten lassen.

Fazit

AS-64 ist ein sehr guter Assembler. Zwar gibt es noch Schwachpunkte (Zeropage-Adressierung, Fehlerquelle schlecht zu finden, kein Insert-Modus), aber das sollte noch zu beheben sein. Auf die noch fehlenden Makros wurde bereits hingewiesen.

Auch die mitgelieferte »Dokumentation« (10 Seiten, DIN A5) ist ein Schwachpunkt. Es wird vorausgesetzt, daß der Benutzer schon Erfahrungen mit vergleichbaren Assemblern besitzt und sie bedienen kann. Ein Anfänger wird völlig alleine gelassen.

Noch ein Wort zum Service: Der Hersteller bietet an, daß man »gegen Einsendung des Moduls mit einem frankierten Rückumschlag kostenlos (!) die aktuellste Version von AS-64 abrufen« kann. Leider ist solcher Service in der Branche noch lange nicht selbstverständlich.

In puncto Geschwindigkeit und Bequemlichkeit beim Assemblieren bleiben kaum Wünsche offen. Setzt man die Kosten (295 Mark) zur Leistung ins Verhältnis, schneidet AS-64 immer noch sehr gut ab. Immerhin erhält man Hardware im Werte von zirka 100 Mark. Das eigentliche Programmpaket ist mit knapp 200 Mark seinen Preis wert. Natürlich gibt es billigere Systeme, kaum aber preiswertere.

(D. Weineck/gk)

Bezugsquelle:
Roßmöller GmbH, Finkenweg 1,
5309 Meckenheim

MAE — ein bewährter Oldtimer

Schon 1978 konnte man den MAE von SM-Software für die damaligen Commodore-Computer kaufen. Seit einiger Zeit ist auch eine C 64 Version auf Diskette erhältlich.

MAE ist eine Abkürzung für Makro-Assembler/Editor. Wie der Name schon sagt, wird ein spezielles Editor-Programm zum Erstellen des Quelltextes benötigt. Dieser Editor arbeitet ähnlich dem Basic-Editor. Es müssen einzelne Zeilen eingegeben werden, die jeweils eine vierstellige Zeilennummer tragen. Diese Zeilennummern bestimmen dann die Reihenfolge der einzelnen Anweisungen.

Natürlich gibt es hier gegenüber dem normalen Basic-Editor erweiterte Möglichkeiten. So ist beispielsweise der Speicherbereich, in dem sich der Quelltext befindet, beliebig vom Benutzer festlegbar. Dies gilt ebenso für den Speicherbereich der Symboltabelle.

Ein eingegebener Quelltext läßt sich beliebig im Quelltextspeicher verschieben, um nachträglich die Reihenfolge von Zeilen zu ändern. Die Zeilennummern können auch, ähnlich einem Renumber, verändert, oder bei der Eingabe automatisch erzeugt werden.

Sehr komfortabel ist hier der Such- und Ersetzbefehl gehalten. Er ermöglichst nicht nur die Angabe von Zeilenbereichen, in denen gesucht werden soll, sondern es kann auch mit Jokerzeichen gesucht werden. Jede gefundene Zeile kann gelistet und dann auf Wunsch verändert oder gelöscht werden.

Die Symboltabelle kann jederzeit aufgelistet, leider aber nicht gespeichert oder geladen werden. Das Floppy-Laufwerk wird mit einigen Befehlen unterstützt, Kassettenbetrieb ist nicht vorgesehen.

Vom Editor kann auch ein Listing des gesamten Quelltextes oder einzelner Zeilennummernbereiche zu einem Drucker gesandt werden.

Der Quelltext kann jederzeit über einen weiteren Befehl formatiert werden. Dabei werden Labels, Opcodes und Kommentare in schön lesbarer Form ausgegeben. Gleichzeitig ist die maximale Länge eines Labels einstellbar.

Assembler-Grundfunktionen

Der eigentliche Assembler arbeitet wahlweise in zwei oder drei Durchgängen (Passes). Im dritten Pass wird ein relatives Lademodul erzeugt, auf das ich noch zu sprechen kommen werde.

Labels können beim MAE maximal 31 Zeichen lang sein. Dies dürfte wohl eindeutige Labelbezeichnungen auch bei sehr langen Programmen ermöglichen. Leider sind die Rechenmöglichkeiten im Quelltext stark eingeschränkt: Es stehen nur Addition und Subtraktion zur Verfügung.

Dem Programmierer steht eine große Zahl von Pseudo-Opcodes zur Verfügung, darunter auch einige, die den Ausdruck eines formatierten Assemblerlistings in Pass 2 steuern. Labels können entweder im Quelltext selbst vor der ihr zugeteilten Anweisung stehen, es ist allerdings auch möglich, Label als intern und extern durch einfache Ausdrücke zu definieren. Externe Label bezeichnen normalerweise Adressen außerhalb, interne Adressen innerhalb des Objektcodes.

Besonders komfortabel ist, daß Labelwerte auch während des ersten Passes eingegeben werden können, so daß man einen Quelltext durch solche Eingaben für verschiedene Speicherbereiche oder Computer nutzen kann.

Auch eine bedingte Assemblierung ist implementiert. Bedingte Assemblierung bedeutet nichts anderes, als daß ein Quelltextblock nur unter bestimmten Bedingungen assembliert wird. Somit ist es leicht, den Objektcode während des Assembliervorganges an bestimmte Spezifikationen anzupassen.

Ein Quelltext kann auch auf seine Syntax getestet werden, während des Assembliervorgangs wird der Objektcode dann nicht gespeichert. Genauso gut ist es möglich, den Objektcode direkt auf Diskette zu leiten, so daß er keinerlei Speicherplatz im Computer benötigt.

Das Modulkonzept

Der MAE geht beim Assemblieren längerer Quelltexte anders vor, als andere Assembler. Wollen Sie einen Quelltext, der länger als der Arbeitsspeicher ist, assemblieren, so können Sie ihn nicht, wie üblich, in mehrere Teile teilen, die sich nacheinander aufrufen. Beim MAE müssen Sie ein Kontroll-Modul definieren, das stets im Speicher vorhanden ist. Dieses lädt dann nacheinander die Quelltextteile in den Speicher und assembliert diese. Das Kontrollmodul besteht ebenfalls aus ganz normalem Quelltext, sollte jedoch zweckmäßigerweise nur die Aufrufe der einzelnen Quelltextteile beinhalten. Dieses Modulkonzept hat den Vorteil, daß Sie einen in mehreren Programmen benötigten Quelltext nur einmal schreiben müssen und dann jedesmal über das Kontrollmodul abrufen können.

Makros

Wichtig bei einem Makroassembler ist natürlich die Makro-Behandlung. Makros dürfen beim MAE beliebig viele Übergabeparameter haben, die dann in die Makrosequenz eingesetzt werden. Auch makrointerne Label sind möglich. Diese müssen mit drei Punkten gekennzeichnet sein. An deren Stelle setzt der MAE beim Assemblieren dann eine Nummer, die von Aufruf zu Aufruf wechselt. So kann es nicht zu einem Fehler aufgrund doppelter Makros kommen. Makros dürfen hier bis zu 32mal ineinander verschachtelt werden.

Sollten Sie einige Makros in großen Programmen ständig benötigen, so müssen Sie ein globales Makro-Modul aufbauen. Dieses steht dann, ähnlich dem Kontrollmodul, ständig im Speicher, so daß jeder Teil des Quelltextes auf das Makro zugreifen kann.

Relative Lademodule

Eine sehr nützliche Einrichtung ist der mitgelieferte Relativlader. Möchte ich einen Quelltext in einem Bereich erzeugen, der schon anderweitig belegt ist, kann ich ihn in einen anderen Speicherbereich assemblieren und dann den dritten Durchlauf starten, der ein relatives Lademodul auf Diskette erzeugt. Dieses Modul kann dann mit einem zusätzlichen Programm, dem Relativlader, an jede beliebige Stelle im Speicher geladen werden. Dabei sind die Speicherbereiche für Programm und Daten frei wählbar. Danach steht das Programm ablauffähig im Speicher und kann absolut gespeichert werden. Der Speichervorgang wird aber vom MAE nicht unterstützt. Dieser Relativlader wird sowohl ablauffähig, als auch selber als relatives Lademodul mitgeliefert. Damit kann der Relativlader an jeder beliebigen Stelle im Speicher stehen. Mit dem MAE ist es nicht möglich, Objektcode direkt zur Diskette zu leiten. Will man längere Objektcodes, die man im Speicher nicht mehr unterbringen kann, erzeugen, muß man einen sehr unkonventionellen Weg gehen. Sie müssen sich Schnittstellen zwischen den einzelnen Programmen definieren, das sind Adressen, über die Daten ausgetauscht werden. Zwei zu verbindende Teilprogramme werden dann durch zweimaligen Aufruf des Relativladers zusammengebunden.

Dokumentation und Sonstiges

Das mitgelieferte Handbuch ist sehr unübersichtlich und unklar. Möchte man zum Beispiel Informationen über den Relativlader haben, stellt man fest, daß dieser an drei verschiedenen Stellen im Handbuch erklärt wird, so daß man ständig vor- und zurückblättern muß. Ebenso verhält es sich mit fast allen Themenbereichen. Das Handbuch einmal in Ruhe durchzulesen ist deswegen kaum möglich.

Schade finde ich es auch, daß bei Fehlermeldungen immer nur Codezahlen und keine Klartexte erscheinen.

Teilweise ist die Bedienung etwas umständlich geraten. Anfängern wird es wohl schwerfallen, sich mit MAE zurechtzufinden, insbesondere mit Blick auf das Handbuch. Aber MAE soll ja schließlich ein Handwerkszeug für professionelle Programmierer sein. Die werden allerdings einige Möglichkeiten anderer Assembler vermissen. Insbesondere stört da die Tatsache, daß man sich zusätzlich zum MAE (Kostenpunkt zirka 140 Mark) noch einen guten Monitor zulegen muß, da man in der Testphase vom MAE allein gelassen wird.

(Boris Schneider/gk)

Bezugsquelle:
SM Software AG, Scherbaumstr. 33, 8000 München 83

T.EX.AS. — mit guter Dokumentation

Ein weiteres Produkt der »Extended« Serie von Interface Age ist der Terminal Extended Assembler, kurz T.EX.AS.

Der T.EX.AS. erfüllt mit einem Programm gleich mehrere Funktionen. Es ist sowohl ein Monitor, ein Direktassembler, ein Reassembler wie natürlich auch ein Makroassembler.

Ein Editorprogramm ist nicht enthalten. Bei T.EX.AS. müssen Quelltexte über den Basic-Editor eingegeben werden. Interface Age schlägt zwar vor, Exbasic Level II zur komfortablen Quelltexteingabe zu verwenden, dies würde aber den Kaufpreis (298 Mark für T.EX.AS.) mehr als verdoppeln und diesen damit wohl über die Kaufkraft vieler Privatanwender stellen.

Der Assembler

In diesem Assembler dürfen Labels bis zu 16 Zeichen haben. Berechnungen sind leider nur auf Addition und Subtraktion beschränkt, was ein nicht unwesentliches Manko dieses Assemblers ist.

Auch ansonsten ist T.EX.AS. nicht gerade mit Pseudo-Opcodes gesegnet. Weder ist eine bedingte Assemblierung vorhanden, noch können Labels umdefiniert werden, was bei Assemblern dieser Preisklasse Standard sein sollte.

Symboltabelle und Quelltext können im zweiten Pass ausgedruckt werden. Der Quelltext wird dabei aber weder formatiert, noch sonst irgendwie verändert, so daß Sie ihn einfacher von Basic aus listen können. Es ist allerdings möglich, den erzeugten Objektcode direkt auf Diskette abzulegen, oder ihn gar nicht auszugeben; dann wird nur die Syntax des Quelltextes überprüft.

Das Einbinden schon geschriebener Quelltextteile ist sehr einfach gelöst. Wird ein solcher Teil benötigt, kann er während des Assembliervorganges nachgeladen und eingefügt werden. Auch hier ist eine modulartige Programmierung wie beim MAE denkbar.

Makros

Auch bei den Makros hapert es ein wenig. Es ist zwar eine beliebige Verschachtelung von Makros erlaubt, (bis auf Selbstaufrufe), aber es gibt keine Möglichkeit, in einem Makro lokale Label zu definieren, ohne sie nicht als Parameter zu übergeben. Also schon eine einfache Schleife in einem Makro, die über einen Labelsprung erfolgen soll, ist äußerst kompliziert und unübersichtlich zu programmieren. Jedes Makro mit internen Labeln dürfte nur einmal aufgerufen werden, soll kein LABEL DECLARED TWICE-Fehler auftreten. In diesen Fällen sollte man bei T.EX.AS. lieber auf Makros verzichten. Dafür ist der Aufbau einer Makrobibliothek sehr einfach, weil jedes Makro als Einzelfile auf der Diskette stehen und bei Bedarf aufgerufen werden kann.

Monitorähnliche Funktionen

In diesem Bereich zeigt T.EX.AS. seine Stärken, da es sich nicht um einen starren Monitor, sondern um eine Art Editiersystem für Objektcode (nicht Quelltext) handelt. Insbesondere die Analyse fremder Programme wird extrem erleichtert.

Als allererstes fällt auf, daß der Bildschirm in zwei Fenster, links und rechts, aufgeteilt ist. Den Cursor kann man mit der Control-Taste vom einen ins andere Fenster springen lassen. Dieses Fensterkonzept ist eine nicht zu unterschätzende Hilfe bei der Programmanalyse. Was machen Sie normalerweise, wenn Sie beim Disassemblieren eines Programmes den Sprung JSR$3000 entdecken? Sie müßten nachschauen, was dort abläuft, um dann später die Stelle mit dem Sprungbefehl wiederzufinden. Beim Disassemblieren mit T.EX.AS. genügt ein Druck auf F6, und schon wird im anderen Bildschirmfenster ab $3000 disassembliert, während im ersten der Programmteil, den Sie gerade untersuchen, stehenbleibt.

Natürlich lassen sich die zwei Fenster auch anders nutzen, zum Beispiel Hexdump im einen und ASCII-Dump im anderen Fenster oder ähnliches.

Selbstverständlich kann in beiden Fenstern unabhängig nach oben und unten gescrollt werden. Aber was bietet T.EX.AS. denn außer diesen Fenstern? Den schon angeführten Dump-Möglichkeiten muß noch die des Adressen-Dumps angefügt werden. Alle Dumps können auch zum Drucker hin erfolgen. Hier wird die Workspace-Konzeption notwendig. Für die meisten Befehle lassen sich Arbeitsbereiche angeben, die in einer Tabelle zwischengespeichert werden. Soll zum Beispiel mehrmals ein bestimmter Bereich disassembliert werden, reicht eine einmalige Definition. Bei anderen Befehlen ist diese Bereichsfestlegung allerdings sinnvoller, zum Beispiel beim Speichern von Bereichen auf Disk. Wenn Sie kurz hintereinander öfters denselben Bereich speichern wollen, weil Sie ihn gerade bearbeiten, müssen die entsprechenden Adressen nicht jedesmal angegeben werden.

Zusätzlich zum Makroassembler ist auch ein Direktassembler vorhanden. Dieser ist insbesondere in der Testphase nützlich. Er kann auch in beschränktem Maße mit Labels arbeiten.

T.EX.AS arbeitet üblicherweise im Dezimalsystem, da man der Meinung war, daß dies einfacher für den Benutzer sei. Unverbesserliche können natürlich auch alle Ein- und Ausgaben ins Hexadezimale umschalten.

Zwei noch erwähnenswerte Kommandos sind der Verschiebe- und der Suchbefehl. Zusätzlich zum normalen Verschieben von Speicherblöcken können Programme einfach an neue Adressen angepaßt werden.

Der Suchbefehl läßt kaum Wünsche offen. Es ist zum Beispiel möglich, im Objektcode alle Befehle zu suchen, die etwas in den Bereich von $3459 bis $7777 schreiben, sich alle indirekten Sprünge heraussuchen lassen, und so weiter. Beim Su-

chen von Kommandos kann der Suchbereich und der Bereich, auf den sich die gesuchten Befehle beziehen, angegeben werden. Außerdem sind Joker für den Befehlscode, das Argument und die Adressierungsart möglich. Leider ist diese Suchroutine nicht auch für den Quelltext verfügbar.

Dem Benutzer stehen zehn adreßdefinierbare Breakpoints zur Verfügung, bei deren Erreichen ein T.EX.AS. Warmstart durchgeführt wird. Dadurch können einzelne Informationen (Akkuinhalt und ähnliches) verlorengehen.

Nur kurz angesprochen werden soll hier der Reassembler, der aus Objektcode wieder Quellcode machen soll. Dieser Programmteil ist mit absoluter Vorsicht zu genießen. Mir ist T.EX.AS. zweimal abgestürzt, als er einen OUT OF MEMORY ERROR gab. Außerdem ist dieser Reassembler außerst unpraktisch. Eine Redefinition von Labels ist nicht vorgesehen.

Trifft der Reassembler auf einen nicht als Opcode definierten Wert, so wird im Quelltext nicht etwa das entsprechende Byte, sondern einfach zwei Fragezeichen ausgegeben. All diese kleinen Macken machen den Reassembler praktisch nutzlos.

Dokumentation

T.EX.AS. ist der wohl einzige Assembler, der auch mit einem, auf ihn abgestimmten, Assemblerlehrbuch ausgeliefert wird. Es ist ziemlich ausführlich und fängt wirklich bei Null an. Auch die Anleitung selbst ist nicht schlecht. An manchen Stellen, die Anfängern Schwierigkeiten bereiten könnten, wird auf die entsprechenden Kapitel im anderen Buch verwiesen. Wenn ich ein ehrliches Fazit ziehen soll, muß ich sagen, daß T.EX.AS zu teuer ist. Mit 298 Mark ist T.EX.AS. zwar ein gutes Lehrsystem, aber professionelle Programmierer, und solche die es werden wollen, werden bei T.EX.AS. einiges vermissen. (Boris Schneider/gk)

Bezugsquelle:
Interface Age Verlag GmbH, Vohburgerstr. 1, 8000 München 21. Preis: 298 Mark

ASSI/M

Mit allen Wassern gewaschen ist der ASSI von D. Zabel.

ASSI/M ist ein ganzes Programmpaket, bestehend aus einem Editor (FSE), dem Assembler (ASM) und einem Monitor (DEMON). Alle drei Programme werden gemeinsam auf einer Diskette geliefert. Es gibt auch eine Version die alle drei Programme vereinigt, dafür aber enorm viel Speicherbedarf hat. Durch die konsequente Aufteilung in Editor, Assembler und Monitor konnte jedes einzelne Programm sehr umfangreich geschrieben werden, ohne daß bei ASSI weniger freier Speicherplatz als bei anderen Assemblern zur Verfügung steht.

Der Editor (FSE)

FSE steht für Full-Screen-Editor, er arbeitet also bildschirmorientiert und ohne Zeilennummern. Aufgrund seines umfangreichen Befehlssatzes gleicht er schon fast einem Textverarbeitungsprogramm. Die vorhandenen Befehle sind jedoch immer am Ziel orientiert, Programme perfekt schreiben und editieren zu können.

Fast der ganze Bildschirm steht zur Eingabe des Quelltextes zur Verfügung. Unten wird er durch eine Statuszeile begrenzt, die stets die Position des Cursors, den Arbeitsmodus und den noch freien Speicherplatz angibt. Nach oben und unten kann mit sehr hoher Geschwindigkeit gescrollt werden. Angefangen beim einfachen Löschen und Einfügen von Buchstaben, Wörtern, Zeilen und ganzen Textteilen, reicht der Befehlssatz über freisetzbare Tabulatoren bis hin zum komfortablen Such- und Ersetzbefehl. Man kann die Suche auf bestimmte Zeilen oder Spalten beschränken, Groß- und Kleinschreibung ignorieren, und Joker beim Suchen und beim Ersetzen (!) angeben.

Der FSE merkt sich automatisch die Stelle, an der die letzte Veränderung im Text vorgenommen wurde. So können Sie schnell mal etwas nachschauen und später mit einem Tastendruck zum Ausgangspunkt zurückkehren. Daß Diskette, Kassette und Drucker voll unterstützt werden, ist da schon fast selbstverständlich. Die Funktionstasten sind belegt, es gibt vom Benutzer frei definierbare Tasten und, und, und...

Es würde den Rahmen dieses Tests sprengen, alle Möglichkeiten, die der FSE bietet, aufzuzählen.

Der Assembler (ASM)

Der leistungsfähige Editor läßt natürlich auf einen ebenso leistungsfähigen Assembler hoffen. Und diese Hoffnungen werden vom ASM voll erfüllt.

Der Quelltext wird entweder von Kassette oder Diskette während des Assemblierens gelesen. Bei Kassettenbetrieb muß deswegen mindestens einmal zurückgespult werden. Dadurch ist allerdings auch eine ziemlich lange Assemblierzeit bedingt. Dafür ist wiederum sehr viel Speicher für den Objektcode und die Symboltabelle verfügbar.

Labels dürfen bis zu 250 Zeichen lang sein. Außerdem gibt es eine Vielzahl von Rechenfunktionen: Alle vier Grundrechenarten, Bitschiebeoperationen, logische Funktionen (AND, OR,XOR,NOT) und logische Vergleiche dürfen beliebig miteinander kombiniert werden. Die Hierarchie wird durch Klammern festgelegt. Angenommen werden Zahlen im Hexadezimal-, Dezimal, Oktal-, und Binärformat sowie ASCII-Zeichen.

Als besonders großer Vorzug muß hier das Block-Konzept erwähnt werden. Teile des Quelltextes können jederzeit als Block definiert werden, und alle in ihnen definierten Labels sind dann lokal. Zum Beispiel können zwei verschiedene Programmteile das Label LOOP verwenden, wenn die eine Sequenz als Block definiert wurde und dort die eine Definition von LOOP erfolgte. Somit kann man jederzeit Quelltext aus anderen Programmen übernehmen und braucht sich keine Sorgen um etwaige doppelt definierte Label zu machen. Auch der Aufbau einer Unterprogrammbibliothek ist kein Problem. Denn neben dem einfachen Verketten von Quelltextfiles, kann man Unterprogramme, die auf Diskette vorhanden sind, mit einem Pseudo-Opcode einbinden. Sie werden dann zur Assemblierung nachgeladen. Die Ausgabe von Listings während des zweiten Pass verläuft hier ähnlich wie bei anderen Assemblern, wahlweise auf Drucker, Bildschirm, Floppy oder mehreren Ausgabegeräten gleichzeitig. Der Objektcode kann entweder beliebig im Speicher plaziert, oder aber auch direkt auf Diskette geschrieben werden.

Natürlich ist auch ein bedingter Assembliermodus vorhanden. Ebenso können beim ASSI Eingaben von Labels während des ersten Pass gemacht werden.

Makros

Der ASM beherrscht natürlich auch Makros. Bei allen Makroaufrufen wird automatisch ein Block um das Makro herum gelegt. So sind die in einem Makro definierten Parameter wieder lokal, das heißt unabhängig von Parametern anderer Programmteile universell verwendbar. Makros dürfen andere Makros oder auch sich selbst aufrufen; letzteres ist jedoch nur bei bedingter Assemblierung sinnvoll, da sonst der Speicher überläuft. Eine Beson-

derheit ergibt sich noch bei den Übergabeparametern. Neben normalen Ausdrücken können auch Strings oder Befehle übergeben werden, die dann in das Makro eingebaut werden.

Und weil die Definition von Makro-Bibliotheken genauso einfach geht, wie die schon oben erwähnte von Unterprogrammen, werden zwei Makrobibliotheken mitgeliefert. Die eine enthält 17 Makros, die hauptsächlich der 16-Bit-Arithmetik dienen, sowie einen User-Stack definieren, auf dem, ähnlich dem normalen Stack, Daten nach dem First in/Last out Prinzip gespeichert werden können.

Die andere ist fast schon sensationell zu nennen, denn sie ermöglicht strukturierte Assemblerprogrammierung. Es stehen dann Befehle wie IF..THEN..ELSE...,REPEAT..EXIT, REPEAT..UNTIL und ähnliches in Assembler zur Verfügung. Da diese Bibliotheken jederzeit erweitert werden können, bilden sie einen idealen Grundstock für jeden Assemblerprogrammierer.

Der Monitor (DEMON)

Und auch das dritte Programm fügt sich nahtlos in das System ein. DEMON ist ein sehr komfortabler Maschinensprachemonitor. Er ist ideal zum Austesten von Programmen geeignet. Schwerpunkte sind die hervorragende Breakpointbehandlung und die Trace-Modi.

Beim Austesten eines Programmes können bis zu fünf normale Breakpoints gesetzt werden. Gelangt der Programmcounter an einen Breakpoint, so wird automatisch in den Einzelschrittmodus des DEMON gesprungen, wo Sie sich den weiteren Verlauf des Programms Schritt für Schritt anschauen können. Zusätzlich ist ein User-Breakpoint definierbar. Sie stellen zum Beispiel den Fehler fest, daß Ihr Programm bestimmte Speicherstellen überschreibt, die wichtig sind. Nun schreiben Sie ein kleines User-Breakpointprogramm, das nach Abarbeitung jedes Befehls angesprungen wird und diese Speicherstellen überwacht. So können Sie in wenigen Minuten Fehler ausfindig machen, an denen Sie sonst Stunden herumknobeln würden. Auch ganz verrückte Abfragen wie: »Wann ist die Summe von X und Y Register kleiner als $93 ?« sind als Userbreakpoint realisierbar. Schon kurz angesprochen wurde der Einzelschrittmodus. Außer dem normalen Abarbeiten und Anzeigen jedes Befehls ist es möglich, Sprünge, die über eine bestimmte Adresse hinaus gehen, direkt ausführen zu lassen. Das spart enorm Zeit, wenn viele ROM-Routinen verwendet werden.

Natürlich sind sämtliche Monitor-Standardfunktionen wie Hex-Dumps, Disassemblieren, Suchen von Bytes und Zeichenketten, Laden und Speichern, Vergleichen und Verschieben von Speicherbereichen (auch mit Adreßangleichung bei Programmen), ein Mini-Assembler, Registeranzeige etc. integriert.

Außerdem können Rechnungen im hexadezimalen, dezimalen und binären Zahlensystem vorgenommen werden, die fast so komfortabel wie im ASM sind. Ein spezieller Suchbefehl findet alle die Zeropage betreffenden Befehle. Damit können auch die vier von DEMON benötigten Zeropageadressen beliebig verändert werden. Überhaupt ist DEMON im Speicher frei verschieblich, so daß Sie niemals mit einem auszutestenden Programm in Konflikt kommen können.

Alles in allem läßt auch DEMON keine Wünsche offen.

Dokumentation

Das Handbuch macht auf mich einen sehr guten Eindruck, da alle Befehle in einer vernünftigen Reihenfolge leicht verständlich und mit einigen Beispielen erklärt werden. Einzig beim FSE wird die logische Reihenfolge durch einen Anhang, der die Erweiterungen mancher Funktionen nachträglich beschreibt, durchbrochen. Allerdings werden auch hier Assembler-Kenntnisse vorausgesetzt. Zusätzlich werden zu allen drei Programmen alle wichtigen Einsprungpunkte und Speicher beschreiben, so daß eine Modifikation durch den Benutzer sehr leicht möglich ist. Die Transparenz der drei Programme, gekoppelt mit ihrer Befehlsvielfalt und Nützlichkeit, machen das Programmpaket ASSI (Preis: 220 Mark) zum fast idealen Assembler. Einzig und allein die geringe Arbeitsgeschwindigkeit kann ein Kritikpunkt sein.

(Boris Schneider/gk)

Bezugsquelle:
Dirk Zabel, Stresemannstr. 50, 1000 Berlin 61

	AS-64	ASSI	MAE	T.EX.AS.
Editor	eigener Full-Screen-Editor	eigener Full-Screen-Editor	eigener Zeileneditor	Basic-Editor
Labels	max. 11 Zeichen	max. 250 Zeichen auch redefinierbare Labels (Variablen)	max. 79 Zeichen	max. 16 Zeichen
Monitor	je nach Version nachladbar oder resident	DEMON wird mitgeliefert	—	viele monitorähnliche Funktionen
Re-Assembler	wird mitgeliefert	—	—	eingebaut
Quelltext Verkettung/Einbindung	Verkettung	Verkettung oder Einbindung	Einbindung über Steuermodul	Einbindung
Objektcode direkt auf Diskette	nein	ja	nein, aber relative Lademodule	ja
Bedingte Assemblierung möglich?	ja	ja	ja	nein
Formatierung des Quelltextes	automatisch bei der Eingabe	bei Ausgabe in Pass 2 auch form. Eingabe möglich	bei Ausgabe in Pass 2	nur durch formatierte Eingabe
Makrobibliothek möglich?	noch nicht vorhanden	ja, zwei werden mitgeliefert	ja, mit globalem Makromodul	Makros direkt von Disk abrufbar
Besonderheiten	EPROM-Modul mit Diskette	Möglichkeit der Blockstrukturierung	Relativ-Lader wird mitgeliefert	zwei unabhängige Bildschirmfenster
Preis	298 Mark	220 Mark	140 Mark	298 Mark

Grundlage dieser neuen Lernidee ist die in Bulgarien und der UDSSR entwickelte Suggestopädie. Diese Methode nutzt als wesentliches Element die Wirkungen eines ganz bestimmten Musikhintergrundes und besonderer Sprachtechniken, die im Zusammenhang mit Entspannungsübungen zu einer tiefen Entspannung und gesteigerter Konzentrationsfähigkeit führen.

In den westlichen Ländern wurde diese Methode mit Elementen der Autosuggestion und Yoga verbunden und erweitert. Das Ergebnis dieser Verknüpfung heißt Superlearning, und ist in den USA schon längere Zeit in der Praxis erprobt worden.

Softlearning

Softlearning baut auf den Erkenntnissen von Superlearning auf, und nutzt zudem die speziellen Eigenschaften des Computers. Die Kosten für diese Kurse sind mit 198 Mark noch erschwinglich.

Die Grundausstattung, die man für diese Lernsoftware benötigt, besteht aus dem Commodore 64, einem Floppy-Laufwerk, einem Audiokassettenrecorder und der »Systemeinheit S«. In ihr ist eine Diskette mit den Programmteilen, die für alle Kurse gleich sind, ein Audio-Adapter und eine Audio-Kassette enthalten. Der Audio-Adapter wird zur Synchronisation zwischen C 64 und Audio-Kassettenrecorder eingesetzt. Die Audio-Kassette der Systembasis S enthält auf der einen Seite Entspannungsübungen, auf der anderen Seite einen Probekurs mit der Plansprache Esperanto.

Der Englisch-Grundkurs

Der Englisch-Sprachkurs besteht aus einer Kursdiskette, vier Audio-Kassetten und einem Lehrbuch. Das Lehrbuch sollte allerdings nur zum Nachschlagen verwendet werden. Die Kassetten sind sowohl für Stereo- als auch für Mono-Recorder geeignet.

Der Kurs beginnt mit einer Entspannungsübung. Zu einer für das Auge angenehmen Grafik wird barocke Musik gespielt, und durch zwei Stimmen werden Hilfen zur Entspannung gegeben. Schon hier wird die Suggestion angewandt. Die barocken Klänge sollen nach Forschungsergebnissen besonders entspannen. Auch Ohren, die bisher nicht an klassische Musik gewöhnt waren, stellen sich sehr schnell darauf ein. Die zwei Stimmen, eine weibliche und eine männliche, sprechen ihren Text in Deutsch und Englisch. Die Entspannungsübung soll zirka eine Woche lang trainiert werden, bevor man mit dem eigentlichen Kurs beginnt.

An die Entspannungsphase schließt sich ein englischer Textteil an, der vorgelesen wird und auf dem Bildschirm erscheint. Auch hier soll man sich an den Klang der Sprache und an das Schriftbild gewöhnen. Ein Verstehen ist vorerst nicht notwendig. Wie auch bei der

Vokabeltraining »Silbenrätsel«

Soft Learning- die weiche Welle des Lernens

Entspannen und Lernen heißt die Devise bei der Softlearning-Reihe von SM, die auf Erwachsenen-Fortbildung ausgelegt ist. Aber auch ältere Schüler ab den Klassen neun bis zehn, dürften diese Software schon sinnvoll einsetzen können.



»Entspannungsgrafik«

Hauptmenü von Softlearning

Entspannungsübung spielt im Hintergrund ständig Musik. Sobald der Text der ersten Lektion erledigt ist, schließt sich eine zweite Entspannungsphase an. Während dieser Übung werden die unbekannten Vokabeln und ihre Übersetzung geliefert.

Ist dieser Kursteil durchlaufen, schließen sich die Vokabel-Trainingsprogramme an. Das Training beginnt mit einem Multiple-Choice-Test. Dies bedeutet, daß man aus mehreren Übersetzungen eines Wortes die richtige zuordnen soll. Während dieses Tests werden als »Erfolgserlebnis« Punkte vergeben. Maximal werden drei Punkte verteilt. Für jeden Fehlversuch bekommt man einen Pluspunkt weniger. Ist man bei null Punkten angelangt, zeigt der C 64 die richtige Lösung an. Minuspunkte werden nicht vergeben, denn auch hier soll der »Entspannungszustand« nicht durch Leistungsdruck beeinflußt werden. Der nächste Punkt im »Softlearning-Lehrplan« ist ein Silbenrätsel. Die deutsche Bedeutung von Vokabeln muß aus vorgegebenen Wortbruchstücken zusammengesetzt werden.

Weitere Programmteile sind Lückentextdiktate, die Übersetzung ganzer Sätze, Grammatik-Übungen und »Zettelkasten-Abfrage«. Dabei werden Vokabeln einer Lektion in zufälliger Reihenfolge abgefragt. Wird eine Vokabel richtig übersetzt, bekommt sie für die interne Verarbeitung eine Markierung. Vokabeln ohne Markierung haben bei der zufälligen Auswahl Priorität, so daß falsch übersetzte Vokabeln nach kurzer Zeit erneut abgefragt werden. Vokabeln die häufig richtig übersetzt worden sind, haben nach diesem Schema soviele Markierungen, daß sie nur sehr selten erneut abgefragt werden. Dieser Aufbau wiederholt sich bei jeder neuen Lektion.

Der Weg, den SM-Software mit dieser neuen Form der Lernsoftware eingeschlagen hat, muß sich in der breiten Anwendung erst noch bewähren. Nach einer kurzen Eingewöhnungszeit kam ich mit diesem System sehr gut zurecht. Besonders gut haben mir die verschiedenen Vokabeltrainingsmöglichkeiten gefallen. Diese Art der Lernprogramme ist in der Erwachsenenfortbildung in Verbindung mit einem Computer sicherlich ein großer Schritt nach vorne. (rg)

Info: SM Software AG, Scherbaumstraße 33, 8000 München 83, 089/6371211

Basic ist out — Es lebe Forth

Viele sind des Basic überdrüssig geworden. Die Sprache Forth ist eine Alternative. Unsere Einführung in Forth beginnt an einem konkreten Beispiel: der Programmierung des Decompilers.

Vor einiger Zeit erhielt ich ein Programm, das sich nach dem Starten mit »FIG-FORTH« meldete. Forth, soviel wußte ich schon, soll eine neue, ungewöhnliche Programmiersprache sein. Nichts Genaues weiß ich nicht. Also gebe ich ein: LIST und RUN und PRINT 5*3 und NEW und ... nichts passierte, außer, daß mein Computer antwortet: »MESS # 0« oder »MESS # 1« oder manchmal auch »OK«. Also erst einmal zur Seite damit ...

Aber diese Sprache läßt mich nicht los — eben weil ich nicht weiß, was ich damit anstellen kann. Bis ein Freund anruft: »Versuch mal VLIST. Ich habe da was gelesen ...« Der Bann ist gebrochen: Nun flitzen plötzlich Worte über den Bildschirm: AND und BASE und BUFFER und CONSTANT und IF und HEX und noch viel mehr, über 200 Worte. Das ist offensichtlich die Befehlsliste von Forth. Und dann bekomme ich die Fotokopie einer Kopie einer Kopie der Kurz-Befehlsbeschreibung eines FIG-Forth-Systems, kaum lesbar, in englisch, aber immerhin. Ich lese die Beschreibung — nein, ich buchstabiere sie beim ersten Mal, zu neu ist das Konzept dieser Sprache. Beim zweiten Lesen fasziniert mich Forth schon; nach dem dritten Lesen steht für mich fest: Basic ist out, Forth ist die Sprache der Zukunft.

Dies war vor etwa einem Jahr. Seitdem ist mir der Aufbau von Forth um einiges klarer geworden. Forth ist, so merkwürdig das klingt, zum großen Teil in Forth geschrieben. Zum Verständnis ist es deshalb wichtig, die Forth-Befehle zu rekonstruieren. Im Computer-Speicher steht aber nicht der Quelltext, sondern das compilierte, also übersetzte Programm. Forth läßt es jedoch zu, den Quelltext aus dem Speicher fast vollständig zu »decompilieren«. Deshalb wird ein interaktiver und einfacher Decompiler vorgestellt, der alle Forth-Worte als Quelltext listet. Das hilft auch bei der Optimierung und der Fehlersuche in eigenen Programmen. Für die folgenden Abschnitte sollte man jetzt seinen Computer einschalten und die — hoffentlich vorhandene — jeweilige Forth-Version laden. Es wird stillschweigend vorausgesetzt, daß man bereits einige erste Tippversuche in Forth hinter sich hat.

So arbeitet Forth

Forth ist eine Programmiersprache, die sowohl einen Interpreter als auch einen Compiler enthält. Nach dem Start eines Forth-Systems ist der Interpreter eingeschaltet. Er wartet auf »Eingangspost« (Input-Stream), die normalerweise von der Tastatur kommt. Diese Eingangspost liest der Interpreter und isoliert daraus das erste »Wort« (WORD), das heißt, eine Zeichenfolge, die durch Leerzeichen begrenzt ist und versucht, dieses Wort auszuführen. Ein solches Wort kann ein Befehl, eine Zahl oder auch eine sinnlose Zeichenkette sein.

Zunächst durchsucht der Interpreter sein »Wörterbuch« (DICTIONARY) nach dem isolierten Wort. Findet er es, so steht in dem zum Wort gehörigen Absatz des Wörterbuches, was zu tun ist. Diese Arbeitsanweisung wird dann von einem Kollegen des Interpreters ausgeführt (EXECUTE). Anschließend kehrt der Interpreter zurück zum Lesen der Eingangspost, isoliert das nächste Wort und blättert wieder im Wörterbuch.

Kann er ein Wort aber nicht finden, so versucht ein anderer Kollege (NUMBER), die Zeichen des Wortes in eine Zahl zu wandeln. Ist das gelungen, so wird die Zahl oben auf den »Stapel« (STACK) gelegt, wo sie für weitere Operationen zur Verfügung steht.

Kann der Interpreter das Wort weder im Wörterbuch finden, noch eine Zahl wandeln, so beschwert er sich mit einer Fehlermeldung, wirft die Zahlen auf dem Stapel und die noch ungelesene Eingangspost weg und wartet auf neue Post von der Tastatur.

Mit diesem interpretierenden Forth könnten wir Programme schreiben, indem wir eine sinnvolle Folge von Befehlen ausführen ließen. Jedoch, jeder Basic-Programmierer weiß, daß Interpretieren, also Lesen des Programmtextes, Nachsehen in einer Befehlsliste und das Wandeln in Zahlen während der Programmausführung lange dauert. Wollen wir schnelle Ausführung haben, so müssen wir unser Programm compilieren.

Im Forth ist der Compiler schon eingebaut. Es gibt einige Worte, deren Ausführung unter anderem darin besteht, den Interpreter aus- und den Compiler einzuschalten. Eins dieser Worte ist der Doppelpunkt »:« (COLON). Es nimmt das nächste Wort aus der Eingangspost und macht daraus den Kopf eines neuen Eintrags im Wörterbuch. Dann wird der Compiler eingeschaltet. Die folgenden Worte der Einganspost werden nun nicht mehr ausgeführt, sondern der Compiler trägt »Zeiger« ins Wörterbuch ein. Diese Zeiger sind 16-Bit-Adressen, also 2 Byte lang und heißen Compilationsadressen. Sie weisen auf die entsprechenden Arbeitsanweisungen im Wörterbuch.

Das Eintragen von Zeigern macht der Compiler so lange, bis er wieder ausgeschaltet wird. Auch für diesen Zweck gibt es Worte: »;« schließt Worte ab, die mit »:« angefangen wurden. Es schaltet den Compiler wieder ab und den Interpreter ein. Im Wörterbuch ist jetzt ein neues Wort eingetragen, dessen Arbeitsanweisung aus einer Liste von Zeigern auf andere Worte besteht.

Ein praktisches Beispiel: Wir wollen von zwei Zahlen auf dem Stapel die oberste behalten, die zweite jedoch wegwerfen. Offenbar erfüllt die Befehlsfolge »SWAP DROP« genau diese Forderung, was der Leser bitte selbst prüfen mag. Wir wollen jedoch ein neues Wort NIP einführen, das unsere Forderung erfüllt:

: NIP (N1 N2 — N2) SWAP DROP;

Wir drücken RETURN, und Forth meldet sich nach dem Compilieren mit »OK«. Nun sehen wir uns an, was der Compiler aus diesem »Kurzbrief« gemacht hat. Dazu dumpen wir den Speicherinhalt mit der Befehlsfolge:

CR HEX ' NIP NFA 0E DUMP
und erhalten das folgende Listing:
83 4E 49 D0 37 58 3F 10 . NIP7X? .
86 F 7D F 33 E 4 44. . . . 3 . . D

Das sieht zunächst etwas wirr aus, doch sind die Bytes bei näherer Betrachtung sinnvoll. Ordnen wir den Speicherdump anders an und ergänzen ihn durch einige Kommentare, dann erhalten wir einen Aufbau wie in Bild 1. Im ersten Byte ist das höchstwertige Bit 7 immer gesetzt, Bit 6 und 5 haben besondere Bedeutung und die Bits 4 bis 0 enthalten die Länge des Namens, hier 3 (Bild 2). Die nächsten drei Byte enthalten dann tatsächlich auch den Namen NIP, nur das Bit 7 des letzten Buchstabens ist wieder gesetzt. Diese 4 Byte bilden das Namensfeld von NIP, gekennzeichnet durch die Namens-Feld-Adresse NFA. Die nächsten beiden Bytes sind das Linkfeld, markiert durch die Link-Feld-Adresse LFA. Im Linkfeld steht ein Zeiger auf das Namensfeld eines vorhergehenden Wortes. Die nun folgenden zwei Bytes bilden das Codefeld, adressiert durch die Code-Feld-Adresse CFA. Im Codefeld befindet sich ein Zeiger auf Maschinencode des jeweiligen Mikroprozessors.

Diese drei Felder bilden den Kopf des Wortes. Das Namensfeld dient dem Interpreter zum Wiedererkennen von »NIP«. Mit Hilfe des Linkfeldes findet der Interpreter zum vorhergehenden Wort, und von dort handelt er sich auf die gleiche Art weiter durch das Wörterbuch. Der

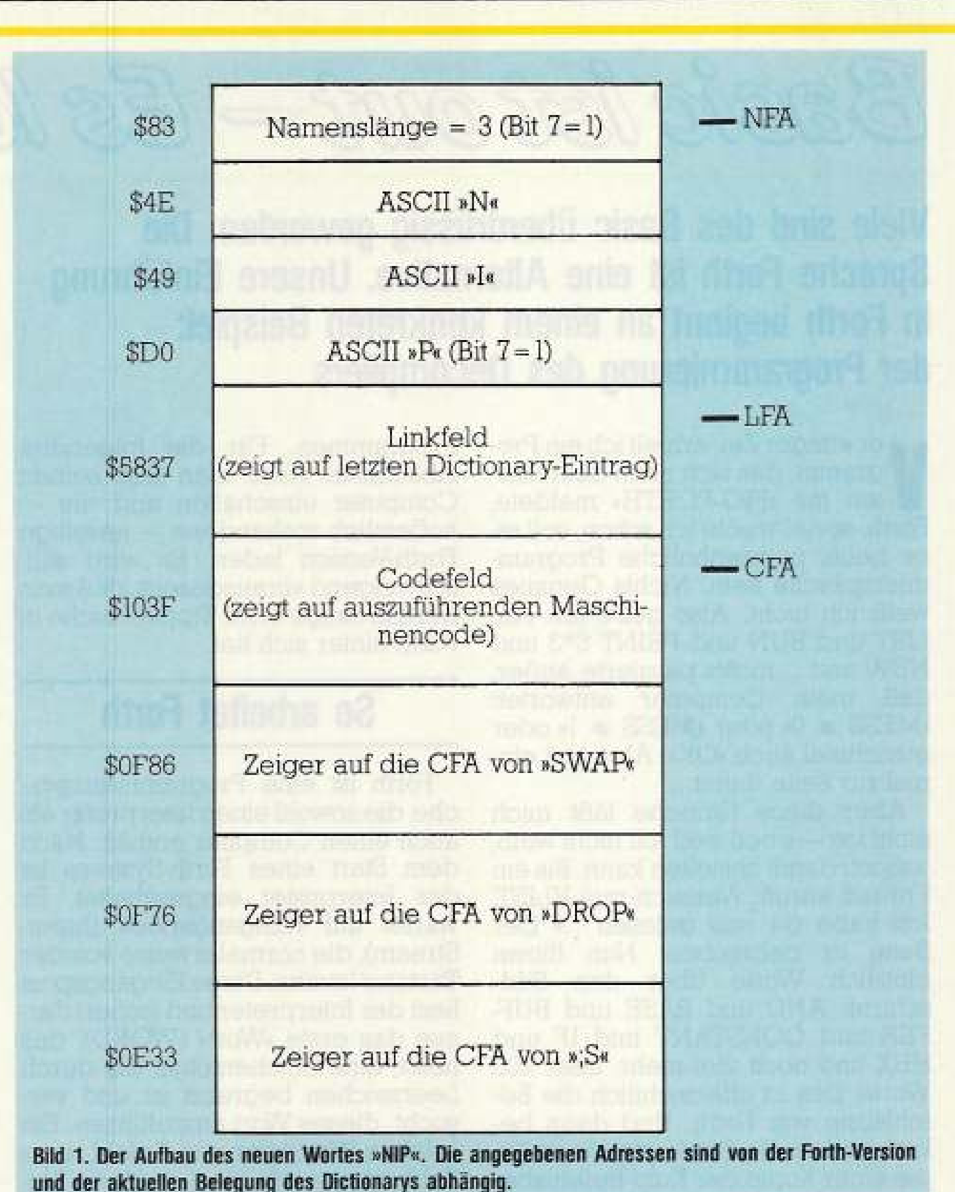


Bild 1. Der Aufbau des neuen Wortes »NIP«. Die angegebenen Adressen sind von der Forth-Version und der aktuellen Belegung des Dictionarys abhängig.

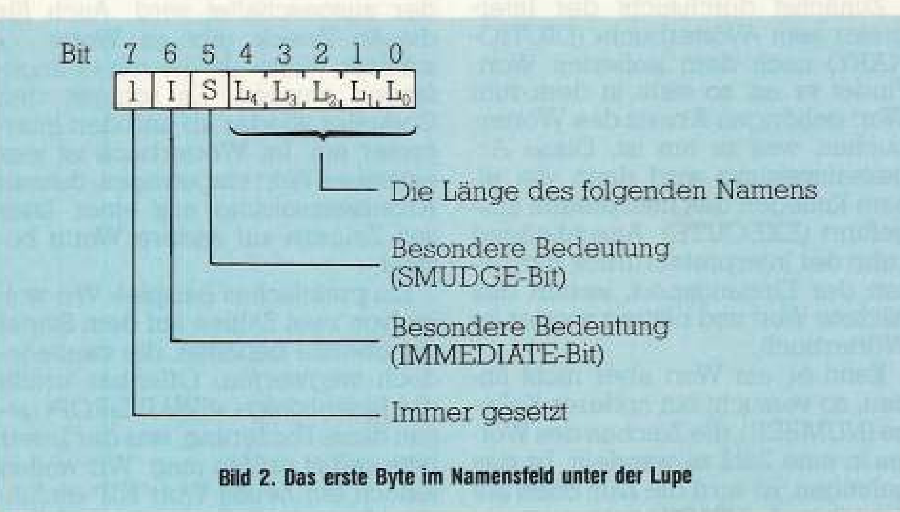


Bild 2. Das erste Byte im Namensfeld unter der Lupe

Inhalt des Codefeldes schließlich bestimmt den Charakter von »NIP«. Es gibt nämlich außer »normalen« Forth-Worten (die durch »:« eingeleitet und mit »;« beendet wurden) noch andere, zum Beispiel »Konstanten« (CONSTANT), »Variable« (VARIABLE) und reine Maschinencodeworte (CODE), wie zum Beispiel »SWAP« und »DROP«. Bei allen »normalen« Worten zeigt der Inhalt des Codefeldes jedoch immer auf denselben kurzen Maschinencode, der bewirkt, daß die im Speicher folgende Liste von Arbeitsanweisungen abgearbeitet wird.

An den Kopf mit seinen drei Feldern schließt sich der Rumpf nahtlos an, beginnend bei der Parameter-Feld-Adresse PFA. Das Parameterfeld ist die Liste von Arbeitsanweisungen und enthält bei unserem Wort NIP drei Zeiger zu 2 Byte; einen auf »SWAP«, einen auf »DROP« und einen Zeiger auf ein Wort »;S«. Bei der Ausführung von NIP beendet das »;S« das Abarbeiten dieser Liste. Wer hat das »;S« ins Wörterbuch geschrieben? Hier hat sich das »;« verewigt, das ja selbst noch einiges mehr tun muß (zum Beispiel den Compiler abschalten). Ganz und gar fehlt die Buchstabenfolge (N1 N2 — N2); dies ist auch gut so, denn mit einer öffnenden Klammer eingeleitete Texte sind Kommentare, der Text wird bis zur schließenden Klammer einfach überlesen.

Die Zeiger auf »SWAP«, »DROP« und »;S« weisen nun nicht auf deren Namensfeld, sondern auf deren Codefeld. Dies beschleunigt die späte-

re Ausführung von »NIP« ganz erheblich, erschwert uns jedoch das Wiedererkennen der Worte beim Speicherdump. Die Zeiger in »NIP« sind nicht nur computerabhängig, sondern auch abhängig von der Forth-Version und von der Stelle, an der »NIP« compiliert wurde. (Deshalb ist der oben gezeigte Speicherdump auch nur ein Beispiel.) Viel komfortabler wäre es, wenn wir statt Zeigeradressen zu dumpen, gleich die Wortnamen listen könnten, aus denen »NIP« sich zusammensetzt. Wir müßten dazu den Zeiger, der zum Beispiel auf die CFA von »SWAP« zeigt, so verbiegen, daß er auf die NFA von »SWAP« weist. Dann könnten wir feststellen, wie lang dieser Name ist und die vier Zeichen SWAP einfach ausdrucken. Diese Aufgabe erfüllt ein Decompiler.

Das Baukasten-Prinzip

Forth ist ein Baukasten, der die wichtigsten Funktionen schon als Bausteine enthält — sie müssen nur noch richtig zusammengesetzt werden. Also basteln wir uns ein Wort, das aus der CFA eines Wortes seinen Namen druckt. Zur Verfügung haben wir ein Wort »ID«, das aus NFA eines Wortes seinen Namen druckt. Das Problem reduziert sich also auf das Verbiegen der CFA auf die NFA, da die Namen verschieden lang sein können, Forth läßt bis 31 Zeichen zu. Auch hierfür gibt es schon ein Bauklötzchen, das verschieden lange Namen berücksichtigt: Habe ich eine PFA, so kann ich sie in die NFA umwandeln mit dem sinnfälligen Wort »NFA«. Unser Problem reduziert sich also weiter auf die Umwandlung einer CFA in die PFA. Das Codefeld enthält immer einen 16-Bit-Zeiger, ist also immer 2 Byte lang. Das Parameterfeld schließt sich immer an das Codefeld an. Also addieren wir zwei zur CFA, auch dafür gibt es ein Klötzchen namens »2+«, und erhalten die PFA, führen dann »NFA« aus und erhalten die NFA; danach drucken wir den Namen mit »ID« und ein trennendes Leerzeichen mit »SPACE«. Diesen neuen Befehl nennen wir »==« (Listing 1).

Nun können wir schon ein bißchen decompilieren: Wir suchen uns die Rumpfadresse von NIP mit:
' NIP
sichern uns die Adresse, holen uns den ersten Zeiger mit:
DUP @
und drucken aus der geholten CFA den zugehörigen Namen mit:
==

```
Screen Nr. 12

0 ( Decompiler, == .NAME N .LIT L                    04.08.84re)
1
2 FORTH DEFINITIONS   HEX
3
4 : ==    ( CFA ---)   2+ NFA ID. SPACE   ;
5
6 : .NAME   ( ADR --- ADR+2)   DUP @ == 2+   ;
7
8 : N   ( ADR --- ADR+2)   DUP U. .NAME QUIT   ;
9
A : .LIT   ( ADR --- ADR+2)   DUP @ . 2+   ;
B
C : L   ( ADR --- ADR+2)   DUP U. .LIT QUIT   ;
D
E --
F
```

Listing 1. Der erste Screen des Decompilers

Auf dem Bildschirm erscheint »SWAP OK«. Jetzt erhöhen wir die Adresse um 2, damit sie auf den nächsten Zeiger weist, mit:
2+
und decompilieren das nächste Wort mit:
DUP @ ==

So könnten wir uns weiter durch NIP oder andere Worte tasten.

Tasten ist hier im doppelten Sinn treffend: Wir müssen viele Tasten drücken und kommen eben deshalb nur langsam voran. Etwas komfortabler muß es schon gehen.

Zunächst fassen wir die immer wieder benötigte Befehlsfolge »DUP @ == 2+« zusammen zum Wort ».NAME« (Listing 1).

Bei ».NAME« sind immerhin fünf Zeichen zu tippen, das ist jedem Forth-Programmierer zuviel. Außerdem hätten wir noch gerne gewußt, wo wir uns im Speicher gerade aufhalten und das ewige »OK« am Ende wollen wir uns beim Decompilieren auch sparen. Die Folge dieser Gedanken ist das Wort »N« aus Listing 1. Das »DUP U.« druckt uns die Adresse, die wir gerade decompilieren (ohne Vorzeichen), das ».NAME« den Namen und das »QUIT« am Ende unterdrückt die OK-Meldung. Es macht aber noch mehr, es bricht die Ausführung von Worten sofort ab, wirft den Rest der Eingangspost weg und kehrt in den Interpreter zurück. Nur der Zahlenstapel bleibt unangetastet.

Nun können wir NIP schon komfortabel decompilieren:
' NIP findet den Rumpf von NIP,
Ausgabe: OK
N decompiliert Namen,
Ausgabe: adr1 SWAP
N decompiliert Namen,
Ausgabe: adr2 DROP
N decompiliert Namen,
Ausgabe: adr3 ;S

Versuchen wir ein anderes Beispiel: Wir wollen uns ein Wort bilden, das zu einer gegebenen Adresse auf dem Stapel genau 1024 hinzuzählt, so daß wir uns in 1-KByte-Schritten durch den Speicher bewegen.
DECIMAL
: 1024+ (ADR — ADR+1024)
1024 + ;

Decompilieren wir nun unser Wort »1024+«, so erhalten wir als ersten Namen die Buchstabenfolge »LIT«, als zweiten Namen irgendwelchen Unsinn, als dritten Namen ein »+« und als vierten Namen ein »;S«. Das »+« und »;S« bedürfen keiner Erklärung, aber woher kommt das »LIT« und das, was sich da als Name nicht decompilieren läßt? Es ist unsere Zahl 1024, die sich durch das vorangestellte »LIT« als 16-Bit-Zahl zu erkennen gibt. Die Zahl selbst steht in den beiden Bytes, die der CFA von »LIT« folgen. Würde beim Compilieren die Zahl einfach nur ins Wörterbuch geschrieben, könnte bei der späteren Ausführung nicht mehr erkannt werden, daß es sich dabei eben nicht um die CFA eines Wortes handelt, sondern um eine Zahl, die auf den Stapel zu legen ist. Die praktische Folge: Stoßen wir beim Decompilieren auf »LIT«, so dürfen wir die nächsten beiden Bytes nicht als Name auffassen, sondern müssen sie als 16-Bit-Zahl einfach ausdrucken.

Basteln wir uns wieder die dazu nötigen Worte, zunächst ein
.LIT (ADR — ADR+2)
das die Adresse, die wir gerade bearbeiten, rettet und die beiden Bytes holt, die unter dieser Adresse zuhause sind. Diese beiden Bytes müssen dann als Zahl (mit Vorzeichen) ausgedruckt und anschließend die Adresse um 2 erhöht werden. Weiterhin ein kurzes Wort
L (ADR — ADR+2)

das ».LIT« benutzt (siehe wieder Listing 1).

Da sich Zahlen von 0 bis 255 in einem Byte darstellen lassen, compiliert Forth solche Zahlen, um Platz zu sparen, mit dem besonderen Wort »CLIT«. Dessen CFA folgt dann die Zahl in nur einem Byte. Auch dafür brauchen wir passende Worte (siehe Listing 2).
.CLIT (ADR — ADR+1)
und
C (ADR — ADR+1)

Nun gibt es noch eine Reihe von Worten, denen bei der Compilation eine nachfolgende 16-Bit-Zahl mitgegeben wird. Das sind die Verzweigungs-Befehle BRANCH, 0BRANCH, (LOOP) und (+LOOP). Die mitgegebene Zahl ist die Sprungadresse; jedoch ist sie nicht absolut angegeben, sondern als Offset zur Adresse, an der der Befehl compiliert wurde.

Schwierig? Wir müssen uns nur den Offset holen und die gerade laufende Adresse, an der wir decompilieren, addieren. Dies macht das Wort:
.BRANCH (ADR — ADR+2)
als Grundlage für:
B (ADR — ADR+2)
(siehe Listing 2).

Noch ein Wort würde uns beim Decompilieren Schwierigkeiten bereiten, wenn wir uns nicht ein »Gegenwort« basteln würden: das ».″«. Ihm folgt ein Forth-String, der zur Ausführungszeit ausgedruckt wird und den wir ebenfalls ausdrucken und die laufende Adresse entsprechend erhöhen müssen. Ein Forth-String ist ähnlich aufgebaut wie das Namensfeld eines Forth-Wortes: Das erste Byte enthält die Länge des Strings, in den folgenden Bytes befinden sich die ASCII-Codes des Strings. Zum Ausdrucken von Strings bietet Forth zwei häufig benutzte Worte an: Haben wir die Adresse, bei der die ASCII-Zeichen des Strings beginnen und wissen wir die Länge, so druckt
TYPE (ADR Länge —)
den String aus. Haben wir jedoch nur die Adresse, unter der wir das Längenbyte eines Forth-Strings finden, so liefert
COUNT (ADR — ADR+1 Länge)
uns die um 1 erhöhte Adresse und das benötigte Längenbyte, so daß ein folgendes »TYPE« richtig vorbereitet ist. Diese beiden Worte benutzen wir in unserem
.STRING (ADR1 — ADR2)
aus Listing 2. Wir dürfen dabei nicht vergessen, die laufende Adresse um die Länge des Strings zu erhöhen, damit wir richtig weiter decompilieren können. Diesem Zweck dient das »2DUP« und das »+« am Ende von ».STRING«. Die Länge des Strings drucken wir uns vor dem String selbst aus. Das bedienerfreundliche Wort S (ADR1 — ADR2) aus Listing 2 benutzt dann wieder ».STRING«.

Werfen wir nochmals einen Blick auf das »CLIT«. Dort finden wir auch unser »COUNT« wieder. Nun geht es bei »CLIT« überhaupt nicht um Strings, das zweckentfremdete »COUNT« erfüllt aber genau das Geforderte: Das Byte unter der laufenden Adresse wird geholt, und die Adresse wird um eins erhöht.

Nun haben wir uns einen einfachen Satz von Worten geschaffen, der sämtliche Forth-Worte interaktiv decompiliert. Dies ist ein erheblicher Fortschritt gegenüber einem Speicherdump. Aber besonders komfortabel ist dieses Decompilieren immer noch nicht. Im zweiten Teil dieses Artikels wird deshalb ein automatischer Decompiler erscheinen, der sämtliche Standard-FIG-Forth Worte erkennt und vollständig decompiliert.

Der automatische Decompiler wird die hier vorgestellten grundlegenden Worte .NAME, .LIT, .CLIT, .BRANCH und .STRING benutzen. die Worte N, L, C, B und S sind durch das in ihnen enthaltene »QUIT« dem interaktiven Betrieb vorbehalten. Das interaktive Decompilieren fördert das Verständnis von Forth jedoch sehr und sollte von keinem Leser ausgelassen werden.

(Georg Rehfeld/ev)

```
Screen Nr. 13

0 ( Decompiler, .CLIT C .BRANCH B .STRING S          10.08.84re)
1
2 : .CLIT   ( ADR --- ADR+1)   COUNT .   ;
3
4 : C   ( ADR --- ADR+1)   DUP U. .CLIT QUIT   ;
5
6 : .BRANCH   ( ADR --- ADR+2)   DUP @ OVER + U. 2+   ;
7
8 : B   ( ADR --- ADR+2)   DUP U. .BRANCH QUIT   ;
9
A : .STRING   ( ADR --- ADR+)   COUNT 2DUP DUP . TYPE +   ;
B
C : S   ( ADR --- ADR+)   DUP U. .STRING QUIT   ;
D
E
F
```

Listing 2. Weitere Befehle zum Decompiler

Das Stapeln ist ein Forth-Prinzip: Bemerkungen zum Stapel.

Betrachten wir den Daten-Stapel etwas näher. Forth-Worte, die Eingangsvariable brauchen oder Ausgangsvariable liefern, benutzen in der Regel den Stapel, um diese Zahlen zu übergeben. Der Additions-Befehl »+« zum Beispiel nimmt die beiden obersten Zahlen vom Stapel, addiert sie und legt die Summe wieder oben auf den Stapel. Dies führt zu einer besonderen Schreibweise von Formeln, der Umgekehrten Polnischen Notation (UPN), wie sie auch bei den Taschenrechnern einer bekannten Firma in Gebrauch ist. Die Basic-Schreibweise »PRINT 2+3« liest sich in Forth: »2 3 + .«

Der Einfluß eines Forth-Wortes auf den Daten-Stapel wird im Quelltext bei jedem Wort als Kommentar angegeben. Das Wort »+« hat demzufolge den Stapel-Kommentar:
+ (N1 N2 — N3)

N1 und N2 sind dabei die Summanden, die vom Stapel genommen werden, die drei Bindestriche deuten die Operation an und N3 ist die Summe, die auf dem Stapel hinterlassen wird. Das oberste Element des Stapels (TOS) steht bei dieser Schreibweise rechts, vor der Ausführung von »+« ist also N2 Stapel-Spitze, danach N3.

Wichtige Befehle zur Stack-Kontrolle sind:
DUP (N — NN) dupliziert TOS
DROP (N —) entfernt TOS vom Stapel
SWAP (N1 N2 — N2 N1) vertauscht die beiden obersten Werte des Stapels
OVER (N1 N2 — N1 N2 N1) kopiert zweites Stapelelement als neuen TOS
ROT (N1 N2 N3 — N2 N3 N1) rotiert die obersten drei Stapelelemente

The Quest

Mit dem C 64 auf der Suche nach dem Drachen



Die Geschichte dieses Grafik-Adventures liegt lange Zeit zurück, an einem Ort weit, weit weg. Eine Geschichte aus vergangener Zeit von Königen, Rittern, Zauberern und Drachen. Eine Geschichte, deren Verlauf Sie als Spieler beeinflussen und verändern.

Ihre Geschichte beginnt im Königreich »Balema«. König »Galt« gibt gerade einer adligen Landsherrin eine Audienz, die ihn bittet, Gnade über Ihr Volk walten zu lassen, das seine Steuern nicht zahlen kann. Schnell stellt sich heraus, daß die Ursache für die Armut ein böser, feuerspeiender Drache ist, der alles zerstört, was ihm in den Weg läuft.

Eine verantwortungsvolle Aufgabe

Der König, weise wie er ist, zeigt Verständnis: Er beauftragt seinen Champion der Kämpfer, den starken Gorn, den Drachen zu suchen und zu vernichten.

Gorn ist zwar groß und stark, hat aber nicht gerade viel Verstand. Das ist auch der Grund, warum Sie ihn begleiten sollen: Sie sind der Berater von Gorn und müssen ihm sagen, wohin er zu gehen hat und was zu tun ist — wenn Gorn nicht gerade starrköpfig ist.

Ein interessanter Spielverlauf

Als erstes sollten Sie sich mit wichtigen Dingen ausrüsten, die Sie auf dem Weg durch das gefährliche Land Balema gut brauchen können. Dazu gibt es einen sogenannten »Provisioner«, bei dem es solche Dinge gibt wie zum Beispiel eine Karte des Landes, eine Lampe, ein Seil, eine Rüstung, etc.

Hat man sich genügend ausgerüstet, dann kann es losgehen. Balema ist ein großes Land, und viele Gefahren treffen dort (und eventuell auch außerhalb des Landes) auf Sie. Es ist immer wieder der Fall, daß hungrige Krokodile oder sogenannte »Lizardmen« Hunger auf Menschenfleisch haben. Auch vor Dieben sollten Sie auf der Hut sein.

Im Verlauf des Spieles werden Sie aber auch auf Freunde treffen, die Ihnen helfen (zum Beispiel der alte Hermit, der sich für eine kleine Gefälligkeit sehr dankbar zeigen kann).

Die Starrköpfigkeit Gorns wird sich übrigens sehr bald zeigen, nämlich sobald eine neue Gefährtin (ein hübsches rothaariges Mädchen) hinzukommt.

Wenn Sie sich nicht dumm anstellen, wird es Ihnen vielleicht sogar gelingen, den Drachen zu finden — und den Grund für seine Zerstörungswut. Was der Grund ist, wollen wir hier natürlich nicht verraten; das soll der Spieler ja schließlich selbst herausfinden.

Versteht ganze Sätze

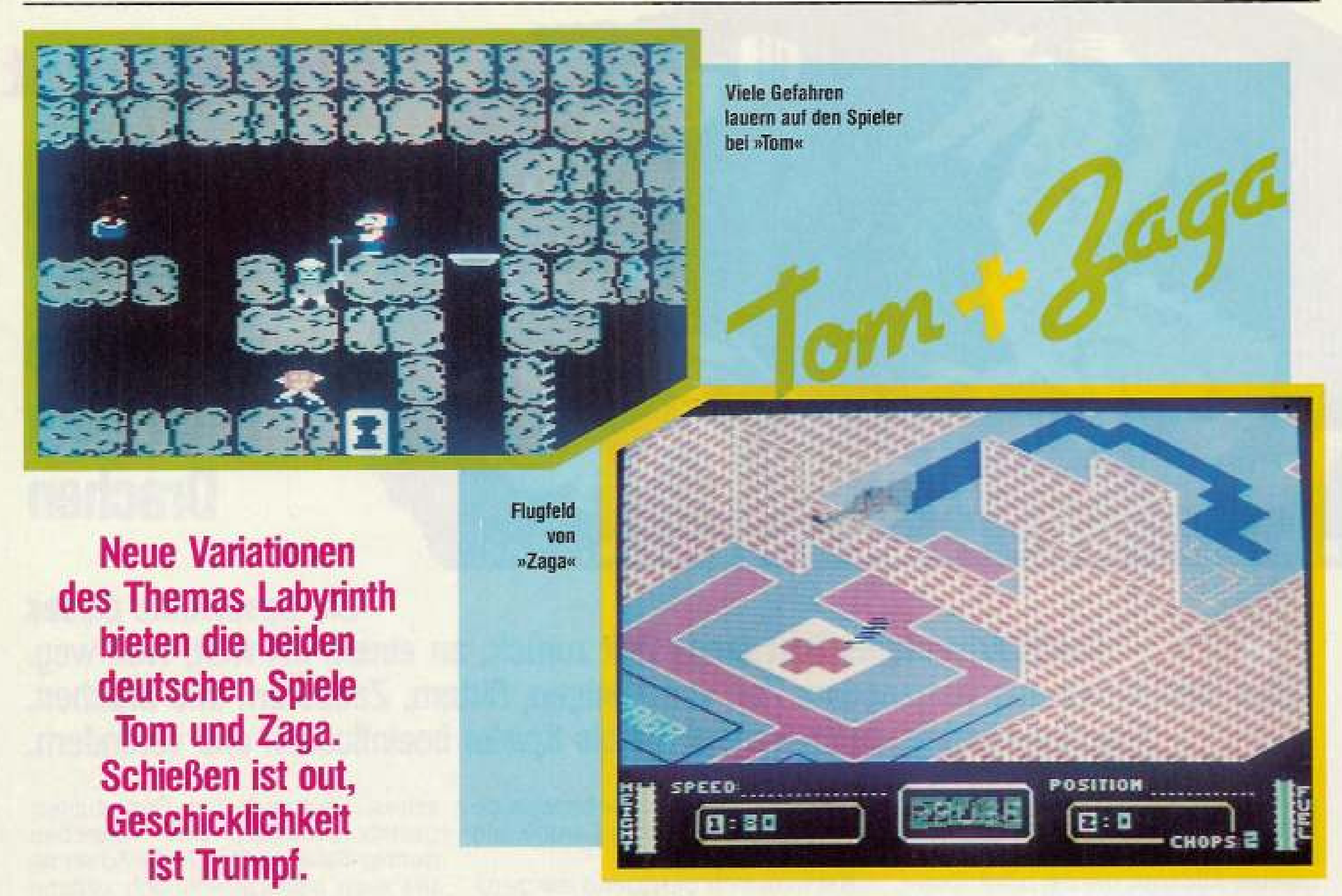
»The Quest« ist ein Grafik-Abenteuerspiel, das ganze Sätze versteht, also einen relativ guten Parser hat (Parser = amerikanisches Fachwort für Sprachinterpretationsroutine). Das Verstehen ganzer Sätze ist für Grafik-Adventures eine außergewöhnlich seltene Sache, da eine solche Routine viel Speicherplatz verbraucht und auch nicht so ohne weiteres gleichzeitig mit der Grafik und dem Hauptprogramm in den Speicher eines kleinen Homecomputers gequetscht werden kann. Das ist wohl auch der Grund, warum der Wortschatz von »The Quest« etwas kleiner geraten ist, so daß man in einigen Stellen des Abenteuers nur durch Raten weiterkommt.

Eine nützliche Option des Spiels ist das An- und Abschalten des Grafikbildschirms. Wer will, kann das Abenteuer auch als reines Textadventure durchspielen. Zumindest wird der Spieler es an den Stellen tun, an denen er schon einmal war und nicht die Ladezeit für die (schon gesehene) Grafik abwarten will.

Recht guter Eindruck

Insgesamt gesehen macht »The Quest« einen recht guten Eindruck, der nur dadurch leicht getrübt wird, daß der Wortschatz dieses Abenteuerspiels an einigen Stellen nur Raten zuläßt. In einer Notenskala zwischen 1 (überdurchschnittlich gut) und 6 (ist sein Geld nicht wert), würde ich »The Quest« die Note 2 (für professionelle Abenteuerspielefans durchaus seinen Preis wert) geben.

(M. Kohlen/aa)



Viele Gefahren lauern auf den Spieler bei »Tom«

Flugfeld von »Zaga«

Neue Variationen des Themas Labyrinth bieten die beiden deutschen Spiele Tom und Zaga. Schießen ist out, Geschicklichkeit ist Trumpf.

Als erstes fallen einem bei beiden Spielen die angenehm kurzen Ladezeiten auf: Obwohl beide Programme sehr lang sind, werden sowohl die Kassette als auch die Diskette sehr schnell geladen. Das Geheimnis heißt Turbolader, das heißt Kassetten werden mit 10facher und Disketten mit 5- bis 6facher Geschwindigkeit in den C 64 geladen.

Bei »Tom« wird der Spieler in die Rolle eines Abenteurers versetzt, der geheimnisvolle Pyramiden nach Schatztruhen durchsuchen soll. Insgesamt sollen 54 Schlüssel zu diesen Truhen in sechs verschiedenen Labyrinthen eingesammelt werden. Dabei sieht der Spieler immer nur einen kleinen Ausschnitt aus einem riesigen Gesamtbild. Sobald sich der Spieler bewegt, wird das Bild in dieser Richtung weitergescrollt. Damit das Ganze nicht zu verwirrend und unübersichtlich wird, gibt es eine Hilfstaste, mit der man sich im Kleinformat das gesamte Labyrinth zeigen lassen kann und erkennt, wo die restlichen Schlüssel verborgen sind.

Natürlich verläuft die Suche nicht störungsfrei: Verschiedene Monster sowie Schlangen, Spinnen und andere feindselige Kreaturen versuchen Tom zu stoppen. Es handelt sich im Prinzip um ein Suchspiel à la Manic Miner oder Jet Set Willy, allerdings mit wirklich riesigen Labyrinthen, die man so schnell nicht alle durchspielt hat.

Die Grafik ist ansprechend und die Musik ist auch nicht übel — wen sie nervt, der dreht einfach leiser.

Auch die Preise können sich sehen lassen: Das Spiel kostet auf Kassette nur 29 Mark, auf Diskette 39 Mark. Tom gibt es übrigens auch in einer leicht gekürzten Version für den VC 20 mit 16 KByte. Praktischerweise befinden sich auf der Kassette beide Versionen, so daß ein VC 20-Besitzer, der zum C 64 aufsteigt, seine Kassette nicht mehr wegzuwerfen braucht. Wie uns der Hersteller mitteilte, wird auch in Kürze eine Version für die neuen Commodore C 16 und C 116 lieferbar sein.

Mit dem Hubschrauber ins Labyrinth

Nach dem Starten von Zaga werden nach dem ersten Bild unwillkürliche Erinnerungen an Zaxxon wach. Allerdings wurde hier der dreidimensionale Eindruck noch etwas verbessert, da die Objekte, die teilweise von Mauervorsprüngen und ähnlichem verdeckt werden, auch wirklich partiell ausgeblendet werden. Beim Spiel selber handelt es sich dann aber nicht um ein Ballerspiel, wie das vielbeachtete Zaxxon eines ist, sondern um ein reines Geschicklichkeitsspiel. Man muß einen Hubschrauber durch ein Labyrinth manövrieren und verschiedenen Hindernissen ausweichen (zum Beispiel beweglichen Mauern). Um den unerfahrenen »Piloten« bei den nicht einfachen Steuermanövern zu helfen, kann auf Wunsch eine Steuerhilfe eingeblendet werden, die anzeigt, in welche Richtung man möglichst fliegen sollte. Das Ganze wird noch durch einen brummigen Hubschrauber-Sound wirkungsvoll untermalt. Dieses Spiel erreicht zwar nicht ganz die Perfektion von Zaxxon, aber erfreulicherweise wird auch dieses Spiel für nur 29 Mark (Kassette) beziehungsweise 39 Mark (Diskette) angeboten, ein Bruchteil dessen, was vergleichbare Spiele aus den USA kosten.

(M. Kohlen/aa)

Bezugsquelle: Kingsoft, Schnackebusch 4, 5106 Roetgen, Tel. 02408/8319

HI-EDDI: ein fantastis Zeichen- und Ma

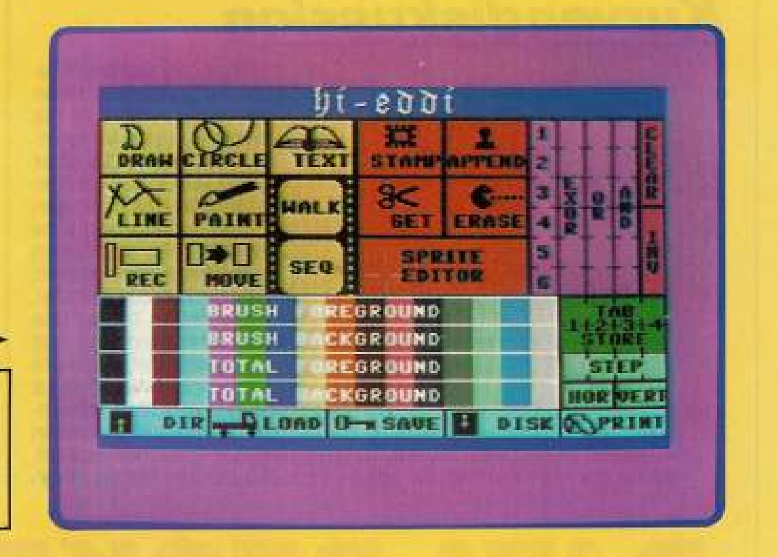
HI-EDDI ist ein High-Resolution-Grafik-Editor, der vieles bietet, was vergeblich sucht. Sehen Sie sich die Bilder an und Sie bekommen



Es lassen sich nicht nur Bilder zeichnen, sondern auch technische Zeichnungen realisieren. Text kann ebenso eingefügt werden wie Sprites und selbstdefinierte Zeichensätze.

Hier eine Aufstellung der herausragenden Eigenschaften:

- Wahlweise als »Schwarzweiß«-Zeichenprogramm oder als farbiges Malprogramm verwendbar, in jedem Fall mit der vollen Auflösung von 320 x 200 Bildpunkten.
- 7 (in Worten: sieben) Schwarzweiß-Bildschirme oder 6 farbige stehen zur Verfügung.
- Integrierter, leistungsfähiger Sprite-Editor (mit Sprite spiegeln, drehen...) und die Fähigkeit, Sprites aus dem Grafikbild herauszukopieren oder einzupflanzen. Damit ist zum Beispiel Ausschnittvergrößerung (Zoom) ebenso wie das Erstellen von »Construction Sets« möglich.
- Mit einer Zeichentrickfilm-Funktion sind fast flimmerfreie Bewegungsabläufe mit bis zu 24 Bildern pro Sekunde möglich.

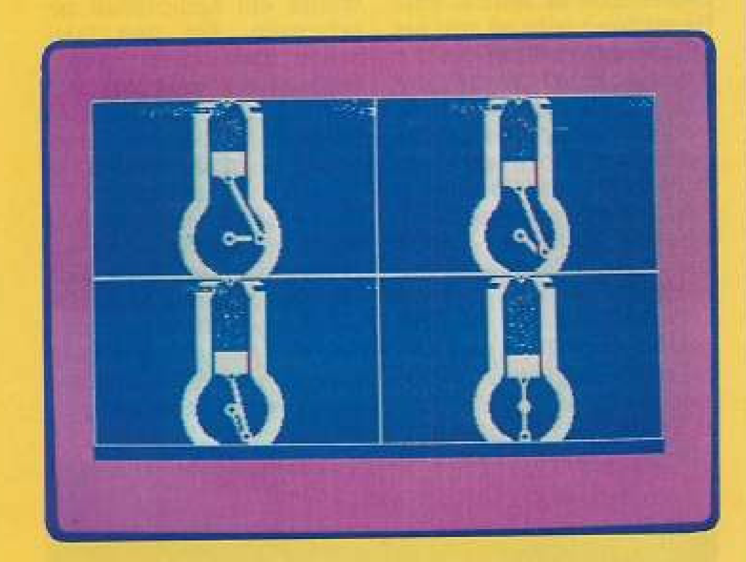


Solche oder auch ganz andere Menüs können Sie sich selbst erstellen. Auf der Diskette zu dieser Ausgabe finden Sie diese und noch andere Beispiele.

ches lprogramm

man bei teuren, kommerziellen Programmen einen Eindruck von seinen Fähigkeiten.

Wenn Sie dieses Bild sechsmal im Speicher ablegen (mit wenigen Tasten möglich) und in jedem Bild die Stellung der Kolben ändern, kann mit dem Walk-Befehl ein Zeichentrickfilm-ähnlicher Effekt erzielt werden (zirka sieben Bilder pro Sekunde). ▼



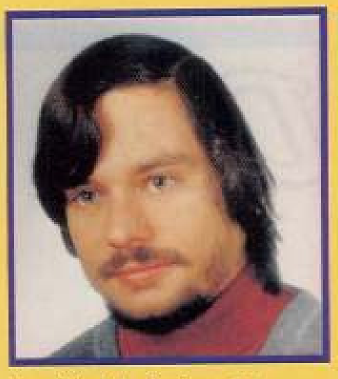
■ Befehlseingabe wahlweise über Tastatur oder eine selbst zu gestaltende Menütafel (à la Koalapainter).

■ Zusammenhängender Ausdruck mehrerer Bilder möglich, das ergibt Superbilder mit einer Breite von minimal 640 Punkten und unbegrenzter Länge (siehe Bild 2 im Listing-Teil).

■ Ferner sind alle »Standardbefehle« wie Draw, Line, Rectangle, Circle, Paint, Move und Text vorhanden.

Natürlich ist klar, daß sowas nur in Maschinensprache geht, und die ist mit rund 4700 DATAs nicht gerade kurz. Gemessen an den Möglichkeiten des Programms ist es allerdings auch nicht viel. Die Bilder auf diesen Seiten lassen die Vielfältigkeit von HI-EDDI nur ahnen. Dabei lassen sich nicht nur mit HI-EDDI erzeugte Bilder laden und bearbeiten, sondern auch Grafikbilder anderer Programme, wie von Koala oder Paint Magic, zum Beispiel die der Dia-Show.

(Hans Haberl/gk)



Hans Haberl, der Gewinner des Listings des Monats, mit HI-EDDI

Ein Chinese verhalf dem Gewinner zu 2000 Mark

Das Programm HI-EDDI ist aus einem Textverarbeitungsprogramm geboren! Diese Aussage verwundert weniger, wenn man an das chinesische »Alphabet« denkt.

Angefangen hat's ganz ohne Computerunterstützung anno 1960 in Steinhöring (also in Bayern). Schon bald begann ich mich für alles zu interessieren, was irgendwie nach Technik roch. Während andere Mumps oder Röteln hatten, litt ich an Radio-Zerlegeritis, Elektronik-Bastleritis und HiFi-Manie. »Dieser Saustall muß ein anderer werden«, hat sich das Christkind wohl gesagt und hat mir anno 79 einen TRS-80 unter den Christbaum gestellt — natürlich nicht ohne vorherige Absprache mit mir. Irgendwann sind mir dann die 16 KByte und die Grafikauflösung von 48 x 128 Punkten zu wenig geworden und es folgte ein C 64. Natürlich hat sich meine Neigung auch in beruflicher Hinsicht ausgewirkt: Seit 1980 studiere ich Elektrotechnik mit Schwerpunkt Datenverarbeitung an der Technischen Universität München. Neben der Computertei ist Musik, besonders klassische, mein Hobby: Brahms, Beethoven, Liszt und Dvorak sind meine Lieblingskomponisten.

Zu dem Programm HI-EDDI wurde ich angeregt, als sich mein chinesischer Freund einen C 64 kaufte: Eines seiner ersten Projekte war es, seine Adresse auf den Bildschirm zu bringen. Nur bewaffnet mit Simons Basic, brauchte er dazu meterlange DATA-Orgien, um die komplizierten chinesischen Zeichen zu definieren. Daraus wurde die Idee geboren, die Schriftzeichen in einem Sprite-Editor zu erstellen und in den Grafikbildschirm einzupflanzen. Die erste Version von HI-EDDI hatte deshalb im wesentlichen nur die Sprite-Befehle. Als chinesisches Textverarbeitungsprogramm eignet sich das schon hervorragend, aber da ich jetzt schon mal mit einem Grafikprogramm angefangen hatte, wollte ich's auch perfektionieren. Nach rund dreimonatiger Arbeit fiel mir dann nichts mehr ein, was ich noch einbauen könnte; das Ergebnis ist die vorliegende Version von HI-EDDI.

(Hans Haberl)

Der C 64 als Handballtrainer

Für eine erfolgreiche Trainingsarbeit ist es unabdingbar, bereits in der Jugendarbeit neben den praktischen Übungen mit und ohne Ball den Spielern taktische Kenntnisse zu vermitteln.



Lebenslauf

Ich bin schon sehr früh — und zwar beruflich — an die »Computerei« geraten. Aus der reinen Anwendung fertiger Programme entsprang sehr bald der Wunsch zu erkennen, was der Computer wirklich kann. Entsprechende Möglichkeiten, die sich mir boten, meine Arbeit mittels Computer zu rationalisieren, nahm ich wahr und kam schon 1969 beim Programmieren zu der Erkenntnis, daß der Computer nur so gut wie die verfügbare Software ist. 1975 brachen meine »hautnahen« Kontakte mit der EDV ab, die ich 1980 mit dem Kauf eines Sharp-PC 1211 wieder auffrischte und 1983 mit dem Kauf eines C 64 krönte. Den Sharp setze ich noch heute für berufliche Zwecke ein, während der C 64 — noch — den rein persönlichen Bereich abdeckt: Telespiele mit gekaufter Software, Führung von Plattendateien, Berechnung von Sportspieltabellen und Entwicklung von Spezialprogrammen, wie »Handballtrainer«, mit eigener Software. Mein kurzfristiges Ziel ist es, einen Personalcomputer für die Unterstützung meiner Planungsaufgaben bei der Montageabwicklung am Arbeitsplatz einzusetzen. Mein Lebenslauf in Kürze:

1943 Geboren in Ostpreußen
1960 Mittlere Reife
1963 Gesellenprüfung
1967 Maschinenbau-Ingenieur
1967 bis 1971 Berechnungs- und Entwicklungs-Ingenieur im Dampfturbinenbau
1969 Programmieren in Basic (Programme für Wärmetechnik und Festigkeit)
1970 Programmieren in Fortran IV (Programme für Strömungstechnik)
1971 bis 1976 Planungs- und Projekt-Ingenieur im Chemie-Anlagenbau
1974 Programmieren in PL I (Anwendung von Montageplanungsprogrammen im Großanlagenbau)
1977 Montageingenieur im Chemie-Anlagenbau
1980 Kauf eines Sharp-PC 1211
1983 Kauf eines C 64

(Manfred Luttkus)

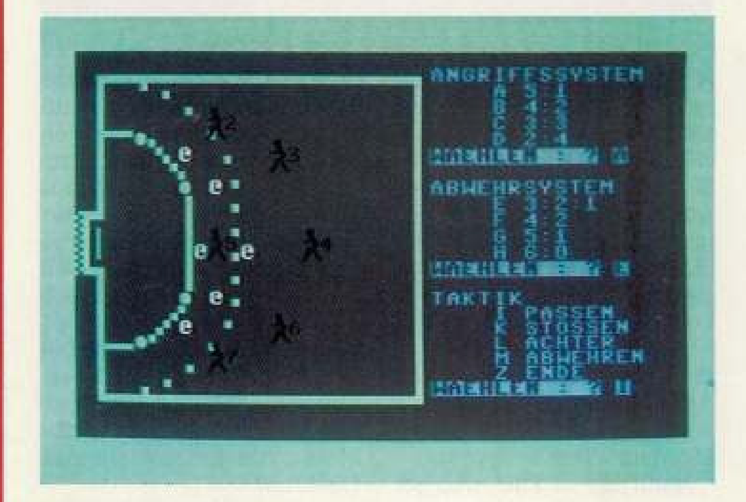
Das Standardgerät hierfür ist die berühmte Kreidetafel, auf der mit mehr oder weniger Geschick des Trainers die Bewegungen des Balles und der Spieler dargestellt werden. Da das Handballspiel nun einmal ein dynamisches und komplexes Spiel ist, sieht so eine Darstellung sehr oft wie ein Schnittmusterbogen aus. An diesem Punkt setzte die Überlegung ein, ob es nicht möglich wäre, hierfür die grafischen Fähigkeiten des C 64 zu nutzen. Zu diesem Zweck wurden einige grundlegende taktische Regeln des Handballspiels ausgewählt.

1. Darstellung der Angriffssysteme 5:1, 4:2, 3:3 und 2:4
2. Darstellung der Abwehrsysteme 5:1, 4:2, 6:0 und 3:2:1
3. Taktische Spielarten wie Passen, Stoßen, Laufspiel (Achterlauf) und Abwehren

Da die Anzahl der darzustellenden beweglichen Objekte beim C 64 auf acht begrenzt ist, aber im realistischen Fall 13 bewegte Objekte erforderlich sind (sechs Angriffsspieler, sechs Abwehrspieler und ein Ball) wurde ein Kompromiß geschlossen. Bei den taktischen Spielweisen Passen, Stoßen und Laufspiel wurden die Angriffsspieler als bewegliche Objekte dargestellt, beim Abwehren jedoch die Abwehrspieler.

Das Programm wurde erstmalig bei der Schulung von A-Jugendlichen eingesetzt. Darüber hinaus wird mit diesem Programm dem interessierten Laien das Handballspiel durchschaubarer und damit wesentlich interessanter gemacht.

(Manfred Luttkus/rg)



Darstellung eines Angriffszuges

Der C 64 als Handballtrainer

Das Handball-Demonstrationsprogramm ist in Basic geschrieben, modular aufgebaut und nutzt vornehmlich die grafischen Fähigkeiten des C 64.

Das Programm ist im wesentlichen in drei Blöcke unterteilt. Im ersten Block sind die DATA-Zeilen abgelegt. Danach folgt der Programmabschnitt mit der Menüauswahl und der Sprite-Aktivierung. Abgeschlossen wird das Programm mit den Unterprogrammen für die einzelnen Programm-Module.

Programmablauf

Nach dem Start des Programmes mit RUN erscheint das Titelbild während des Einlesens der DATA-Zeilen. Anschließend wird das halbe Spielfeld mit dem normalen Zeichensatz dargestellt und auf der rechten Seite des Bildschirmes das Menü angeboten. Aus den drei Hauptgruppen
Angriffssystem (Auswahlelemente A—D),
Abwehrsystem (Auswahlelemente E—H) und
Taktik (Auswahlelemente I—M)
wird jeweils ein Element ausgewählt. Nach dieser Programmauswahl werden die beweglichen Spieler (beim Passen, Stoßen und Laufspiel sind es die Angriffsspieler), der Ball und die statischen Spieler (beim Abwehren sind es die Angriffsspieler) auf dem Bildschirm dargestellt. Entsprechend der gewählten Taktik wird dann die Spielart Passen, Stoßen, Achter oder Abwehren demonstriert. Das Programm wird durch das Drücken einer beliebigen Taste unterbrochen und zum Menü zurückgeführt. Anhand der Darstellungen können die Vorzüge der einzelnen Abwehr- und Angriffssysteme begreifbar gemacht, sowie die Aufgaben der einzelnen Spieler auf den verschiedensten Positionen klar beschrieben werden.

Programmlisting

Aus speichertechnischen Gründen (die Sprites sind im Bereich 15872 — 16834 abgelegt) mußte im Programmlisting auf kommentierende Zeilen verzichtet werden. Entsprechende Erläuterungen sind im nachfolgenden Abschnitt angegeben. Zum besseren Verständnis:

Spielfeldplan für Programmzeile 4000-4120 in Bild 1, Grundaufstellungsplan für Programmzeile 36000-36084 in Bild 2.

(Manfred Luttkus/rg)

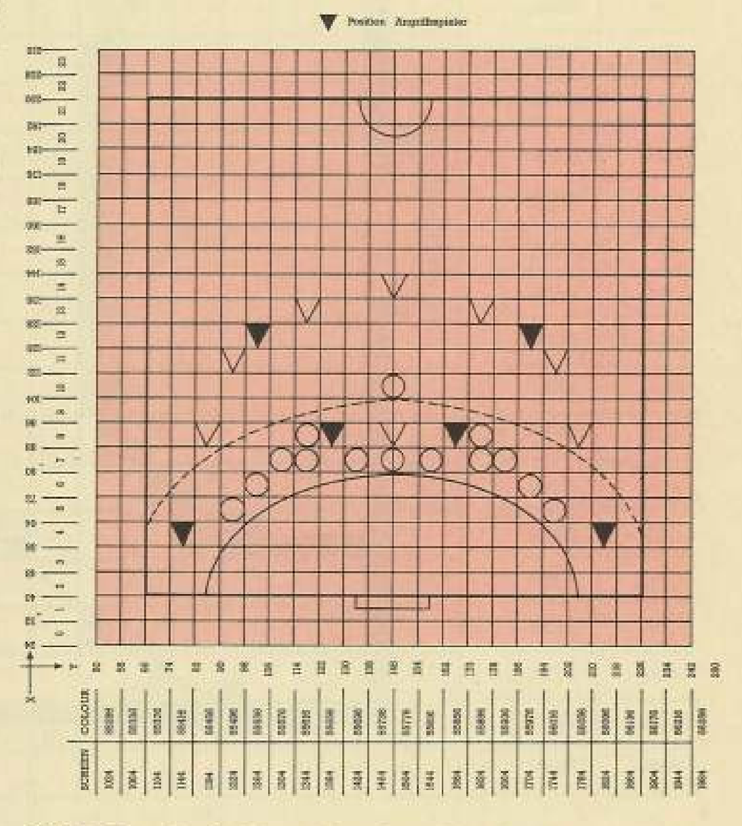


Bild 2. Grundaufstellung, Abwehrspieler (5 : 1)

Listing: »Handballtrainer«

```
10 REM " HANDBALL20 VOM 04.8.84 "                    <203>
20 DIM ZESP%(1,7),DS%(1010),PO%(5,9)                  <188>
1000 DATA 1,2,2,4,4,5,5,4,4,3,3,1,1,0,0,1            <042>
1010 DATA 0,1,1,2,2,0,1,2,2,0,0,1,2,0,0,1,1,2,3,
   5,5,4,4,3,4,3,3,5,5,4,5,4,4,3,3,5                  <150>
3000 DATA 1,1,1,1,1,1,1,1,1,1,-1,-1,-1,-1,1,1,1,
   1,2,3                                              <021>
3010 DATA-1,-1,-1,-1,1,1,2,2,2,2,-2,-2,-1,-1,-1,
   -1,-1,-1,1,1                                       <138>
3020 DATA-3,-2,-2,-2,-2,-1,1,1,1,1,-1,-1,-1,-1,
   -1,-1,-1,-1,-1,-1                                  <072>
4000 DATA 1,1,108,1,22,123,22,22,126,22,1,124
                         <201>
4010 DATA 98,97,226,225                               <212>
4020 DATA 10,0,220,11,0,220,12,0,220,13,0,220
                         <196>
4030 DATA 10,1,226,11,1, 72,12,1, 72,13,1,
   98                                                 <095>
4040 DATA 5,2,120,5,3,120,5,4, 81,5,5,123,6,5,
   124                                                <139>
4050 DATA 6,6,123,7,6,124,7,7,123,8,7,81,9,7,
   225                                                <183>
4060 DATA 10,7,225,11,7,225,12,7,225,13,7,225,
   14,7,225                                           <213>
4070 DATA 15,7, 81,16,7,126,16,6,108,17,6,126,
   17,5,108                                           <192>
4080 DATA 18,5,126,18,4, 81,18,3,121,18,2,
   121                                                <002>
4090 DATA 2,4,124,2,6,123,3,7,108,5,9,126,6,10,
   123                                                <038>
4100 DATA 8,10,124,9,10,108,11,10,124,12,10,108
                         <149>
4110 DATA 14,10,124,15,10,108,17,10,126,18,9,123
                         <220>
4120 DATA 20,7,124,21,6,126,21,4,108              <157>
20010 DATA 0,504,63,567,126,630,189,693,252,756,
```

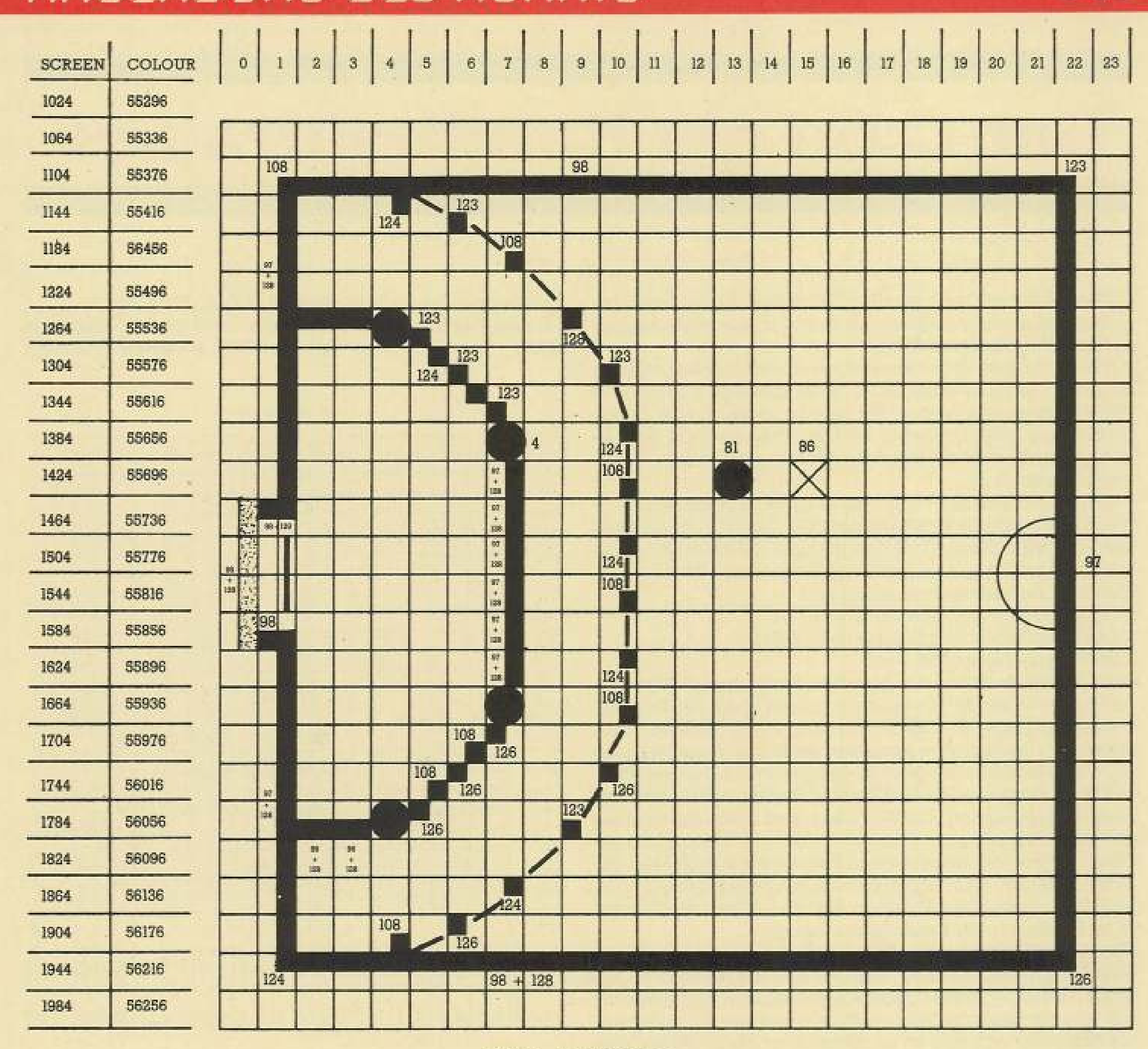


Bild 1. Spielfeldplan

Listing »Handballtrainer« (Fortsetzung)

```
   315,819,378,882,441,945                          <206>
20020 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0                                      <121>
20022 DATA 0,60,0,0,126,0,0,255,0,0,255,0,0,255,
   0,0,255,0,0,126,0,0,60,0                         <105>
20024 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0      <065>
20030 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0,
   0,28,0,0,24,0                                    <045>
20032 DATA 8,24,0,24,56,0,8,124,0,8,102,0,8,67,
   0,8,64,0,28,96,0                                 <250>
20034 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0,
   0,0                                              <112>
20040 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0,
   0,28,0,0,24,0                                    <055>
20042 DATA 28,24,0,34,56,0,2,124,0,4,102,0,8,99,
   0,16,96,0,62,96,0                                <100>
20044 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0,
   0,0                                              <122>
20050 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0,
   0,28,0,0,24,0                                    <065>
20052 DATA 62,24,0,4,56,0,8,124,0,28,102,0,2,67,
   0,2,64,0,60,96,0                                 <046>
20054 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0,
   0,0                                              <132>
20060 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0,
   0,28,0,0,24,0                                    <075>
20062 DATA 4,24,0,12,56,0,20,124,0,36,102,0,62,
   67,0,4,64,0,4,96,0                               <098>
20064 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0,
   0,0                                              <142>
20070 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0,
   0,28,0,0,24,0                                    <085>
20072 DATA 62,24,0,32,56,0,60,124,0,2,102,0,2,
   67,0,34,64,0,28,96,0                             <162>
20074 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0
   0,0                                              <152
20080 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0
   0,28,0,0,24,0                                    <095
20082 DATA 12,24,0,16,56,0,32,124,0,60,102,0,34
   67,0,34,64,0,28,96,0                             <017
20084 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0
   0,0                                              <162
20090 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0
   0,28,0,0,24,0                                    <105
20092 DATA 62,24,0,2,56,0,4,124,0,8,102,0,8,67,
```

```
   0,8,64,0,8,96,0                              <252>
20094 DATA 0,56,0,0,28,0,0,30,0,0,24,0,0,12,0,0,
   0,0                                          <172>
20100 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0                                  <201>
20102 DATA 0,60,0,0,126,0,0,255,0,0,255,0,0,255,
   0,0,255,0,0,126,0,0,60,0                     <185>
20104 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0        <145>
20110 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <186>
20112 DATA 0,28,16,0,52,48,0,246,16,3,6,16,0,6,
   16,0,60,16,0,60,56                           <163>
20114 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <059>
20120 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <196>
20122 DATA 0,28,56,0,52,68,0,246,4,0,6,8,0,6,16,
   0,60,32,0,60,124                             <120>
20124 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <069>
20130 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <206>
20132 DATA 0,28,124,0,52,8,0,246,16,0,6,56,0,6,
   4,0,60,4,0,60,120                            <118>
20134 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <079>
20140 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <216>
20142 DATA 0,28,8,0,52,24,0,246,40,0,6,72,0,6,
   124,0,60,8,0,60,8                            <086>
20144 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <089>
20150 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <226>
20152 DATA 0,28,124,0,52,64,0,246,120,0,6,4,0,6,
   4,0,60,68,0,60,56                            <195>
20154 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <099>
20160 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <236>
20162 DATA 0,28,24,0,52,32,0,246,64,0,6,120,0,6,
   68,0,60,68,0,60,56                           <007>
20164 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <109>
20170 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,
   0,0,56,0,0,24,0                              <246>
20172 DATA 0,28,124,0,52,4,0,246,8,0,6,16,0,6,
   16,0,60,16,0,60,16                           <161>
20174 DATA 0,54,0,0,102,0,0,99,0,0,193,128,0,
   128,128,0,0,0                                <119>
30002 POKE 53281,11:POKE 53280,13
   :PRINT CHR$(147)                             <185>
30005 FOR IH=0 TO 9:PRINT CHR$(17);:NEXT IH
   :PRINT TAB(8)"H A N D B A L L D E M O"       <204>
30008 FOR I=1 TO 8:FOR J=0 TO 1:READ ZB%(I,J)
   :NEXT J:NEXT I                               <143>
30009 FOR I=0 TO 5:FOR J=0 TO 5:READ ZS%(I,J)
   :NEXT J:NEXT I                               <161>
30020 FOR I=0 TO 5:FOR J=0 TO 9:READ PO%(I,J)
   :NEXT:NEXT                                   <015>
30030 PRINT CHR$(147):II=40:FOR XG=1 TO 4       <007>
30035 READ Y,X,CH:POKE 1024+Y*II+X,CH
   :POKE 55296+Y*II+X,13:NEXT XG:READ CH        <203>
30040 FOR XG=2 TO 21:POKE 1064+XG,CH
   :POKE 55336+XG,13:NEXT XG:READ CH            <203>
30045 FOR XG=2 TO 21:POKE 1046+XG*II,CH
   :POKE 55318+XG*II,13:NEXT XG:READ CH         <076>
30050 FOR XG=2 TO 21:POKE 1904+XG,CH
   :POKE 56176+XG,13:NEXT XG:READ CH            <219>
30055 FOR XG=2 TO 21:POKE 1025+XG*II,CH
   :POKE 55297+XG*II,13:NEXT XG                 <013>
30060 FOR XG=1 TO 48:READ Y,X,CH
   :POKE 1024+Y*II+X,CH:POKE 55296+Y*II+X,13
   :NEXT XG                                     <229>
30065 FOR I=0 TO 7:FOR J=0 TO 1:READ ZESP%(J,I)
   :NEXT J:NEXT I                               <108>
30066 FOR I=0 TO 1007:READ DS%(I):NEXT I        <027>
30068 FOR IH=0 TO 21:PRINT CHR$(17);:NEXT IH
                    <191>
30069 FOR IH=0 TO 21:PRINT CHR$(145);:NEXT IH
                    <242>
30070 PRINT TAB(23)"ANGRIFFSSYSTEM"
   :PRINT TAB(27)"A 5:1":PRINT TAB(27)"B 4:2"
                    <137>
30076 PRINT TAB(27)"C 3:3":PRINT TAB(27)"D 2:4"
   :GOSUB 55000:X$=Q$:PRINT                     <135>
30080 PRINT TAB(23)"ABWEHRSYSTEM"
   :PRINT TAB(27)"E 3:2:1":PRINT TAB(27)"F 4:2"
                    <110>
30086 PRINT TAB(27)"G 5:1":PRINT TAB(27)"H 6:0"
   :GOSUB 55000:Y$=Q$:PRINT                     <154>
30090 PRINT TAB(23)"TAKTIK":PRINT TAB(27)"I
    PASSEN":PRINT TAB(27)"K STOSSEN"            <250>
30120 PRINT TAB(27)"L ACHTER"
   :PRINT TAB(27)"M ABWEHREN"
   :PRINT TAB(27)"Z ENDE"                       <217>
30135 GOSUB 55000:Z$=Q$:PRINT                   <101>
30140 IF Z$<>"M"THEN GOSUB 60000:SIT=1:A=-8
                    <118>
30150 IF Z$="M"THEN GOSUB 60100:SIT=0:A=8       <035>
30160 V=53248:N=0:FOR IH=0 TO 6:N=N+2↑IH:NEXT IH
   :POKE V+21,N                                 <099>
30165 FOR IH=0 TO 5:NRSP=2+IH:GOSUB 40000
   :NEXT IH                                     <187>
30170 NRSP=0:IH=6:SX(IH)=SX(4)+A:SY(IH)=SY(4)
   :GOSUB 40000                                 <033>
30175 IF Z$="M"THEN GOSUB 60000                 <202>
30185 IF Z$<>"M"THEN GOSUB 60100                <135>
30186 FOR IH=0 TO 5:POKE SC(IH),96:NEXT IH      <093>
30190 FOR IH=0 TO 5                             <048>
30192 IF SY(IH)=134 THEN SY(IH)=138             <075>
30194 SC(IH)=SX(IH)/8+5*SY(IH)+812
   :SZ(IH)=SC(IH)+54272                         <002>
30195 IF SIT=0 THEN POKE SC(IH),0:POKE SZ(IH),0
                    <147>
30196 IF SIT=1 THEN POKE SC(IH),0:POKE SZ(IH),1
                    <150>
30197 NEXT IH                                   <125>
30200 IF Z$="I"THEN GOSUB 44000                 <225>
30210 IF Z$="K"THEN GOSUB 42000                 <236>
30220 IF Z$="L"THEN GOSUB 50000                 <246>
30230 IF Z$="M"THEN GOSUB 38000                 <007>
30240 IF Z$="Z"THEN GOSUB 63000                 <252>
31000 GOTO 30069                                <028>
36000 SX(0)=88:SX(1)=120:SX(2)=136:SX(3)=88
   :SX(4)=120:SX(5)=88                          <161>
36004 SY(0)=74:SY(1)=90:SY(2)=134:SY(3)=134
   :SY(4)=178:SY(5)=194:RETURN                  <165>
36010 SX(0)=88:SX(1)=128:SX(2)=88:SX(3)=88
   :SX(4)=128:SX(5)=88                          <145>
36014 SY(0)=74:SY(1)=106:SY(2)=114:SY(3)=154
   :SY(4)=162:SY(5)=194:RETURN                  <214>
36020 SX(0)=56:SX(1)=112:SX(2)=136:SX(3)=88
   :SX(4)=112:SX(5)=56                          <173>
36024 SY(0)=66:SY(1)=82:SY(2)=134:SY(3)=134
   :SY(4)=186:SY(5)=202:RETURN                  <176>
36030 SX(0)=56:SX(1)=120:SX(2)=88:SX(3)=88
   :SX(4)=120:SX(5)=56                          <139>
36034 SY(0)=66:SY(1)=90:SY(2)=114:SY(3)=154
   :SY(4)=178:SY(5)=202:RETURN                  <186>
36050 SX(0)=72:SX(1)=88:SX(2)=80:SX(3)=104
   :SX(4)=88:SX(5)=72                           <114>
36054 SY(0)=178:SY(1)=162:SY(2)=134:SY(3)=134
   :SY(4)=106:SY(5)=90:RETURN                   <254>
36060 SX(0)=72:SX(1)=88:SX(2)=80:SX(3)=80
   :SX(4)=88:SX(5)=72                           <079>
36064 SY(0)=178:SY(1)=162:SY(2)=146:SY(3)=122
   :SY(4)=106:SY(5)=90:RETURN                   <008>
36070 SX(0)=64:SX(1)=80:SX(2)=80:SX(3)=104
   :SX(4)=80:SX(5)=64                           <120>
36074 SY(0)=186:SY(1)=162:SY(2)=134:SY(3)=134
   :SY(4)=106:SY(5)=82:RETURN                   <018>
36080 SX(0)=64:SX(1)=80:SX(2)=80:SX(3)=80
   :SX(4)=80:SX(5)=64                           <085>
36084 SY(0)=186:SY(1)=170:SY(2)=146:SY(3)=122
   :SY(4)=98:SY(5)=82:RETURN                    <245>
38000 IG=1:XZ=XB(IG)-16:YZ=YB(IG)-16            <207>
38020 ZY=INT(RND(1)*10):N=PO%(IG,ZY)            <139>
38040 X1=XB(IG):X2=XB(IG+N):Y1=YB(IG)
   :Y2=YB(IG+N)                                 <168>
38050 GOSUB 60400:GOSUB 60450:QB=Q:UB=U:NB=NT
                    <073>
38070 X1=VX(5-IG-N):X2=XB(IG+N):Y1=VY(5-IG-N)
   :Y2=YB(IG+N)                                 <160>
38080 GOSUB 60400:GOSUB 60450:QA=Q:UA=U         <071>
38100 X1=XZ:X2=VX(5-IG):Y1=YZ:Y2=VY(5-IG)       <074>
38110 GOSUB 60400:GOSUB 60450:QZ=Q:UZ=U         <151>
38140 VS=4:NB=20:FOR IN=0 TO NB STEP VS         <183>
38150 XP=XB(IG)+INT(QB*IN/NB)-8                 <022>
38160 YP=YB(IG)+INT(UB*IN/NB):GOSUB 60200       <002>
```

Listing »Handballtrainer« (Fortsetzung)

```
38180 XP=VX(5-IG-N)+INT(QA*IN*0.5/NB):X1=XP
                              <153>
38190 YP=VY(5-IG-N)+INT(UA*IN*0.5/NB):Y1=YP
                              <171>
38200 X=2*(5-IG-N):Y=X+1:GOSUB 60250     <171>
38210 XP=XZ+INT(QZ*IN/NB):YP=YZ+INT(UZ*IN/NB)
                              <229>
38230 X=2*(5-IG):Y=X+1:GOSUB 60250       <208>
38250 NEXT IN                            <024>
38280 XZ=X1:YZ=Y1:IG=IG+N                <055>
38290 GET A$:IF A$=""GOTO 38020          <211>
38300 RETURN                             <191>
40000 POKE 2040+IH,248+IH:POKE V+39+IH,1
   :IF SIT=1 THEN POKE V+39+IH,0         <039>
40040 J=ZESP%(SIT,NRSP):S1=15872+IH*64:S2=S1+62
                              <230>
40060 FOR KH=S1 TO S2:POKE KH,DS%(J):J=J+1
   :NEXT KH                              <028>
40080 X=2*IH:Y=2*IH+1:XP=SX(IH):YP=SY(IH)
   :GOSUB 60250:RETURN                   <004>
42000 ZZ=0                               <074>
42020 IF(X$="D"OR X$="B")THEN GOSUB 43900  <242>
42030 IF(X$="C"OR X$="A")THEN GOSUB 43950  <255>
42040 FOR XG=XA TO XE:X1=XT(XG):X2=XT(XG+1)
   :Y1=YT(XG):Y2=YT(XG+1):GOSUB 60400    <073>
42060 FOR IN=0 TO NT STEP VS:GOSUB 60450:XP=XP-8
   :GOSUB 60200:NEXT IN:NEXT XG          <212>
42130 ZZ=30:GOSUB 60000                  <188>
42150 FOR XG=5 TO 0 STEP-1:X=2*XG:Y=X+1  <227>
42160 IF XG=3 GOTO 42190                 <236>
42170 IF(XG=2 AND X$="B")OR(XG=2 AND X$="D")GOT
   O 42190                               <146>
42180 FOR JJ=0 TO 5:ZQ=ZZ*JJ/5:XP=SX(XG)+ZQ
   :YP=SY(XG):GOSUB 60250                <030>
42185 XP=XB(5)+ZQ-8:YP=YB(5):GOSUB 60200:NEXT JJ

                              <228>
42190 NEXT XG                            <147>
42210 N=1:ZS=ZZ                          <004>
42220 FOR XG=5 TO 1 STEP-1               <241>
42230 IF XG=3 GOTO 42380                 <051>
42240 IF(XG=2 AND X$="B")OR(XG=2 AND X$="D")GOT
   O 42380                               <218>
42244 IF XG=4 THEN N=2                   <146>
42248 IF(XG=4 AND X$="B")OR(XG=4 AND X$="D")THE
   N N=3                                 <054>
42250 X=2*XG:Y=X+1:XN=2*(XG-N):YN=XN+1   <027>
42260 FOR I=0 TO ZZ STEP VS              <015>
42270 XP=SX(XG)+ZS-I:YP=SY(XG):GOSUB 60250
   :XX=SX(XG)+ZS-8                       <105>
42280 XP=XX-I:GOSUB 60200:NEXT I         <143>
42300 QB=SX(XG-N)-SX(XG)+ZZ-ZS
   :UB=SY(XG-N)-SY(XG)                   <021>
42310 FOR I=0 TO ZZ STEP VS              <065>
42330 X=XN:XP=SX(XG-N)-I+ZZ:Y=YN:YP=SY(XG-N)
   :GOSUB 60250                          <003>
42350 XP=XX+INT(QB*I/ZZ)-ZZ:YP=YB(XG)+INT(UB*I/
   ZZ):GOSUB 60200:NEXT I:N=1:ZS=0       <202>
42380 NEXT XG                            <082>
42382 FOR IH=0 TO 5:NRSP=2+IH:GOSUB 40080  <095>
42384 FOR JH=0 TO 20:NEXT JH:NEXT IH     <203>
42390 GET A$:IF A$=""GOTO 42000          <224>
43000 RETURN                             <045>
43900 XA=0:XE=2:VS=4                     <038>
43920 XT(0)=XB(0)+ZZ:YT(0)=YB(0):XT(1)=XB(1)+ZZ
   :YT(1)=YB(1)                          <149>
43921 XT(1)=XB(1)+ZZ:YT(1)=YB(1)         <206>
43930 XT(2)=XB(4)+ZZ:YT(2)=YB(4):XT(3)=XB(5)+ZZ
   :YT(3)=YB(5):RETURN                   <127>
43950 XA=0:XE=3:VS=4                     <089>
43970 XT(0)=XB(0)+ZZ:YT(0)=YB(0):XT(1)=XB(1)+ZZ
   :YT(1)=YB(1)                          <199>
43980 XT(2)=XB(2)+ZZ:YT(2)=YB(2):XT(3)=XB(4)+ZZ
   :YT(3)=YB(4)                          <227>
43990 XT(4)=XB(5)+ZZ:YT(4)=YB(5):RETURN  <233>
44000 XA=0:XE=4:VS=5                     <141>
44020 FOR XG=XA TO XE                    <075>
44025 X2=XB(XG+1):X1=XB(XG):Y2=YB(XG+1)
   :Y1=YB(XG):GOSUB 60400                <227>
44060 FOR IN=0 TO NT STEP VS:GOSUB 60450:XP=XP-8
   :GOSUB 60200:NEXT IN:NEXT XG          <172>
44110 XA=5:XE=1:VS=5                     <254>
44120 FOR XG=XA TO XE STEP-1             <053>
44125 X2=XB(XG-1):X1=XB(XG):Y2=YB(XG-1)
   :Y1=YB(XG):GOSUB 60400                <074>
44160 FOR IN=0 TO NT STEP VS:GOSUB 60450:XP=XP-8
   :GOSUB 60200:NEXT IN:NEXT XG          <016>
44205 GET A$:IF A$=""GOTO 44000          <000>
44250 RETURN                             <020>
50000 GOSUB 36030:SX(1)=SX(1)+16:SX(4)=SX(4)+16
                              <000>
50010 FOR IG=0 TO 5:X=2*IG:Y=X+1:XP=SX(IG)
   :YP=SY(IG):GOSUB 60250:NEXT IG        <101>
50050 XP=SX(1)-8:YP=SY(1):GOSUB 60200    <049>
50100 FOR JJ=0 TO 2:FOR TT=1 TO 8        <102>
50110 X1=SX(ZB%(TT,0))-8:X2=SX(ZB%(TT,1))-8
   :Y1=SY(ZB%(TT,0)):Y2=SY(ZB%(TT,1))    <049>
50115 GOSUB 60400                        <013>
50120 IF(TT=1)OR(TT=5)THEN GOSUB 50500   <176>
50130 FOR L=0 TO 8:XP=X1+Q*L/8:YP=Y1+U*L/8
   :GOSUB 60200                          <207>
50140 IF(TT=1)OR(TT=5)THEN GOSUB 50600   <197>
50150 NEXT L                             <119>
50160 NEXT TT:NEXT JJ                    <045>
50200 GET A$:IF A$=""GOTO 50100          <129>
50300 RETURN                             <206>
50500 JS=0:IF TT=5 THEN JS=3             <005>
50510 FOR K=0 TO 2:X=JS+K:Y=2*JJ:Z=Y+1
   :S1=ZS%(X,Y):S2=ZS%(X,Z)              <128>
50515 XX(K)=SX(S1):YY(K)=SY(S1)          <081>
50520 QQ(K)=SX(S2)-SX(S1):UU(K)=SY(S2)-SY(S1)
   :NEXT K:RETURN                        <104>
50600 FOR K=0 TO 2:X=(JS+K)*2:Y=X+1      <110>
50610 XP=XX(K)+QQ(K)*L/8:YP=YY(K)+UU(K)*L/8
   :GOSUB 60250:NEXT K:RETURN            <037>
55000 PRINT TAB(23);CHR$(18);
   :INPUT"WAEHLEN : ";Q$:RETURN          <040>
60000 IF X$="A"THEN GOSUB 36000          <181>
60010 IF X$="B"THEN GOSUB 36010          <193>
60020 IF X$="C"THEN GOSUB 36020          <205>
60030 IF X$="D"THEN GOSUB 36030          <217>
60040 FOR IB=0 TO 5:XB(IB)=SX(IB):YB(IB)=SY(IB)
   :AX(IB)=SX(IB):AY(IB)=SY(IB):NEXT IB  <038>
60050 RETURN                             <010>
60100 IF Y$="E"THEN GOSUB 36050          <035>
60110 IF Y$="F"THEN GOSUB 36060          <047>
60120 IF Y$="G"THEN GOSUB 36070          <059>
60130 IF Y$="H"THEN GOSUB 36080          <071>
60140 FOR II=0 TO 5:VX(II)=SX(II):VY(II)=SY(II)
   :NEXT II                              <010>
60150 RETURN                             <110>
60200 POKE V+12,XP:POKE V+13,YP:RETURN   <179>
60250 POKE V+X,XP:POKE V+Y,YP:RETURN     <207>
60400 Q=X2-X1:U=Y2-Y1:NT=INT(SQR(Q*Q+U*U))
   :RETURN                               <082>
60450 XP=X1+INT(Q*IN/NT):YP=Y1+INT(U*IN/NT)
   :RETURN                               <002>
63000 END                                <142>
```

Listing »Handballtrainer« (Schluß)

Aufbau des Programms Handballtrainer

1000-1010	Daten für Spielbewegung »Achter«
3000-3020	Daten für Spielbewegung »Abwehren«
4000-4120	Daten für Spielfeld
20010-20174	Daten für Abwehr- und Angriffs-Sprites
30002-30005	Bildschirmfarben und Titelbild
30008-30020	Daten einlesen für Spielbewegung »Achter« und »Abwehren«
30030-30060	Daten einlesen für Spielfeld-Darstellung
30065-30066	Daten einlesen für Abwehr- und Angriffs-Sprites
30068-30135	Menü und Menüauswahl
30140-30197	Grundstellung für Abwehr- und Angriffs-Sprites
30200-31000	Verzweigung zu den Programmodulen
36000-36084	Standardkoordinaten der Angriffs- und Abwehr-Sprites
38000-38300	Programm-Modul »Abwehren«
40000-40080	Sprites-Daten auswählen und Sprites »einschalten«
42000-43990	Programm-Modul »Stoßen«
44000-44250	Programm-Modul »Passen«
50000-50160	Programm-Modul »Achter«
55000	Auswahlroutine
60000-60150	Unterprogramm für die Grundstellung der Angriffs- und Abwehr-Sprites
60200-60450	Unterprogramm für die Sprite-Position
63000	Programmende

HI-EDDI, ein fantastisches Zeichen- und Malprogramm

Wie mächtig HI-EDDI ist, können Sie aus der umfangreichen Funktionsbeschreibung ersehen. Es kann sogar mit den »professionellen« Programmen verglichen werden, auf jeden Fall ist es das beste Grafikprogramm zum Abtippen.

Nach dem Start meldet sich HI-EDDI mit der Frage »Betriebsart«. Wird hier 0 eingegeben (oder einfach RETURN), wird HI-EDDI als »Schwarzweiß«-Programm betrieben, bei 128 dagegen als farbiges Malprogramm. Weitere mögliche Eingaben werden später behandelt, ebenso die Befehlseingabe mittels Menütafel. Zunächst werden alle Befehle, die im folgenden zusammengestellt sind, durch Tastendruck eingegeben.

D Draw — »Freihändig« zeichnen

Im Draw-Modus kann mit dem kreuzförmigen Cursor (der wird mit einem Joystick in Port 2 gesteuert) »freihändig« gezeichnet werden. Bei gedrücktem Feuerknopf werden die überfahrenen Punkte gesetzt. Um Punkte zu löschen, muß zusätzlich die SHIFT-Taste (oder SHIFT-LOCK zum Feststellen) gedrückt werden.

L Line — Linien ziehen

Mit dem ersten Knopfdruck am Joystick wird der Anfangspunkt einer Linie festgelegt, mit dem zweiten der Endpunkt, mit dem dritten wieder ein Anfangspunkt etc.. Der jeweilige Anfangspunkt wird auf F7 gespeichert (siehe Funktionstasten), damit lassen sich Strahlen besonders einfach zeichnen. SHIFT wie bei D.

R Rectangle — Rechtecke zeichnen

Wie L — auch bezüglich SHIFT und F7 —, jedoch wird zwischen den zwei markierten Punkten ein Rechteck gezeichnet.

C Circle — Kreise zeichnen

Der erste Knopfdruck ergibt den Mittelpunkt (auf F7 gespeichert, damit ist einfaches Zeichnen konzentrischer Kreise möglich), der zweite einen beliebigen Randpunkt, von dem aus HI-EDDI im Uhrzeigersinn einen Kreis zieht. Wird beim Anstoßen an den Rand abgebrochen.

Möchte man einen Ausschnitt eines Kreises, der nicht ganz auf den Bildschirm paßt, zeichnen, so muß der zweite Knopfdruck einen Punkt am Bildschirmrand markieren, von dem aus der sichtbare Kreisausschnitt im Uhrzeigersinn gezeichnet werden kann. Allerdings darf der Radius maximal 256 Punkte betragen.

P Paint — Ausmalen begrenzter Flächen

Cursor mitten auf die auszumalende Fläche setzen und Knopf drücken. Sollte durch ein Loch in der Umrandung der ganze Bildschirm vollzulaufen drohen, so kann der Vorgang durch nochmaligen Knopfdruck abgebrochen werden (dazu ist allerdings eine gute Reaktion nötig, denn HI-EDDI füllt Flächen um einiges schneller als Simons Basic). Zum Löschen von Flächen: Bildschirm mit »I« invertieren, entstandenes »Loch« vollaufen lassen, zurückinvertieren.

M Move — Verschieben von Bildschirmbereichen

Mit den ersten beiden Knopfdrücken werden zwei diagonale Ecken des zu transportierenden Bereiches gesetzt (nach dem zweiten Knopfdruck erscheint eine farbliche Markierung dieses Bereiches), der dritte Knopfdruck gibt die linke, obere Ecke des Zielbereiches an. Der Zielbereich muß noch ganz auf den Bildschirm passen (sonst wird der Knopfdruck nicht akzeptiert), er darf jedoch den Quellbereich überlappen oder in einem anderen Bildschirm liegen. Die Auflösung des Move-Befehls entspricht der des normalen Textbildschirmes, es kann also nur im 40 x 25-Raster verschoben werden.

Soll ein Bereich mehrmals kopiert werden, so muß er nicht jedesmal neu markiert werden: Ein Druck auf die Pfeil-nach-oben-Taste holt die letzte Markierung wieder auf den Bildschirm, es kann danach sofort der Zielbereich bestimmt werden (funktioniert nur, solange zwischenzeitlich kein Moduswechsel erfolgte).

Hat man sich beim Markieren vertan, so löscht ein Druck auf die Pfeil-nach-links-Taste die Markierung wieder (gilt auch für L, R, C).

T Text — Buchstaben und Grafiksymbole einfügen

Es erscheint ein 8 x 8-Pixel großer Rahmen, der sich fast so benimmt, wie der Blinkcursor im Textbildschirm: Drucken von Buchstaben und Grafikzeichen, Cursorsteuerung mittels Cursortasten, Löschen mittels DEL (rückwärts) und INST (vorwärts, um für Text Platz zu schaffen), Reverse on/off und Umschaltung der beiden Zeichensätze mit C = SHIFT (es können alle 512 Zeichen der beiden Zeichensätze gleichzeitig dargestellt werden!).

Daneben bleibt die Cursorsteuerung mittels Joystick erhalten, ebenso alle anderen Befehle, die jedoch nur durch gleichzeitiges Drücken der CTRL-Taste eingegeben werden können. Durch Anwahl eines anderen Modus, zum Beispiel CTRL D für Draw, wird der Textmodus verlassen und es ist wieder »Ein-Hand-Eingabe« der Befehle möglich.

G Get Sprite — Sprite aus Bildschirm kopieren

In diesem und den folgenden drei Modi erscheint ein spritegroßer Rahmen als Cursor. Auf Knopfdruck wird der Bildschirmausschnitt, auf dem der Rahmen sitzt, in das Sprite hineinkopiert, anschließend geht HI-EDDI automatisch in den Append-Modus, das Sprite kann an anderer Stelle wieder eingepflanzt oder im Sprite-Editor bearbeitet werden.

A Append — Sprite in Bildschirm einfügen

Auf Knopfdruck wird der Spriteinhalt in den Bildschirm eingefügt, ohne jedoch den Bildschirmausschnitt vorher zu löschen (Oder-Verknüpfung). Bei gleichzeitiger Bewegung wird das Sprite zum »programmierbaren Pinsel«.

S Stamp — Sprite auf Bildschirm kleben

Wie A, jedoch wird vor dem Einfügen der Untergrund gelöscht, das Sprite wird wie eine Briefmarke auf den Bildschirm geklebt.

E Erase — Löschen

Der Rahmen wird zum Radiergummi, der alles löscht, was er überfährt.

F Foreground-Colourmode — Vordergrund einfärben

B Background-Colourmode — Hintergrund einfärben

Diese beiden Modus-Befehle sind nur wirksam, wenn HI-EDDI als farbiges Malprogramm betrieben wird. Doch dazu ist vorweg einiges zum Konzept zu sagen:

Im Gegensatz zu den meisten käuflichen Programmen, die im Multicolour-Modus arbeiten (mehrere Farben, aber nur halbe Auflösung), ist HI-EDDI konsequent als Zeichenprogramm mit maximaler Auflösung konzipiert. Die Farbfähigkeiten sind

nur ein »Nebenprodukt« und deshalb nicht so ausgeprägt: Pro 8 x 8-Punkte-Feld (entsprechend einer Position im Textbildschirm) stehen nur zwei Farben — je eine für Vorder- und Hintergrund — zur Verfügung. Bei insgesamt 1 000 Feldern und 16 Farben kann das allerdings auch recht bunt werden. Außerdem ist es durch die Trennung von Zeichnen und Einfärben möglich, bestehende »Schwarzweiß-Bilder« leicht nachträglich zu colorieren.

Nun zu den Befehlen **F** und **B**: Sie schalten einerseits die Rahmenfarbe (und die Farbe des Sprite-Inhalts bei **A** und **S**) weiter, andererseits wählen sie den Fore- beziehungsweise Back-Mode an, in dem auf Knopfdruck die gesetzten Pixel (=Vordergrund) beziehungsweise gelöschten Pixel (= Hintergrund) des 8 x 8-Feldes, auf dem sich der Cursor befindet, mit der Rahmenfarbe eingefärbt werden. Beispiel: Um den Vordergrund gelb anzumalen, muß die Taste F so oft gedrückt werden, bis der Rahmen gelb ist. Dann kann mit dem Cursor gepinselt werden.

Hat man ein Feld zuviel angepinselt, so kann durch gleichzeitiges Drücken der SHIFT-Taste (wie bei D, L, R, C) die gerade gemalte Farbe gelöscht werden. Genauer gesagt: Es erscheint wieder die Farbe, die beim letzten Bildschirmwechsel dort war. Als Bildschirmwechsel gelten:

1. Bildschirmspeicherwechsel (1 bis 7)
2. Ausschalten des High-Resolution-Bildschirmes (Sprite-Editor, Befehle H, V, SHIFT W, Disk- und Druckerbefehle).
3. Move-Befehl, auch wenn nur innerhalb eines Bildes »gemovet« wurde.

Die Pfeil-nach-links-Taste (Korrekturtaste) hat die gleiche Wirkung wie SHIFT + Knopfdruck, jedoch für den ganzen Bildschirm: Es werden alle Farbveränderungen seit dem letzten Bildschirmwechsel rückgängig gemacht.

Direkte Befehle

Alle bisher beschriebenen Befehle haben eins gemeinsam: Sie dienen zur Anwahl eines Modus, sie haben also eine »Nachwirkung«, vor allem, was die Funktion des Feuerknopfes anbelangt. Die nun folgenden Befehle ändern den gerade eingestellten Modus nicht, sie haben nur eine unmittelbare Wirkung.

1 bis 7 beziehungsweise 1 bis 6: Bildschirmspeicher-Anwahl

HI-EDDI hat im Schwarzweiß-Betrieb sieben und im Farbbetrieb sechs Bildschirmspeicher, die durch Eingabe ihrer Nummer auf den Bildschirm geholt werden.

Wozu so viele? Beispiele: Abspeichern von verschiedenen Zwischenstadien der bearbeiteten Werke, bei denen man wieder ansetzen kann, wenn man was vermurkst hat. Oder Erstellen von »Construction Sets«: Aus einer Anzahl zum Beispiel von Schaltsymbolen kann man mittels der Sprite-Befehle im Nu Schaltpläne aufbauen. Für die Befehle **Print** und **Walk** sind sieben Speicher eigentlich zuwenig, aber im C 64 haben eben nicht mehr Platz!

I Invertieren

Die Pixel des Bildschirmes werden invertiert. In Zusammenhang mit den Befehlen Und, Or, Exor, Append, Stamp, Get, Erase, Paint und Print ergibt das eine Vielzahl von Möglichkeiten.

Un Und-Verknüpfung

Der aktuelle (= sichtbare) Bildschirm und der Bildschirm Nummer n werden Und-verknüpft, das Ergebnis im aktuellen Bildschirm abgelegt.

On Oder-Verknüpfung

Wie U, jedoch Oder-Verknüpfung. Eignet sich zum Beispiel zum Duplizieren von Bildern. Da die Befehle I, U, O, X keinen Einfluß auf die Farbinformation eines Bildes haben, wird sie bei Farb-Betrieb nicht dupliziert. Soll dies geschehen, so muß mit dem Move-Befehl dupliziert werden.

Xn Exor-(Exclusiv-Oder-)Verknüpfung

Zweimalige Exor-Verknüpfung mit demselben Bild bewirkt, daß dieses wieder aus dem sichtbaren Bild »herausgefieselt« wird.

F, B (Fore, Back) — Rahmenfarbe weiterschalten
SHIFT F Total Foreground — Vordergrund-Farbe
SHIFT B Total Background — Hintergrund-Farbe

Die Befehle F und B wurden bereits bei den Modusbefehlen behandelt. Im Schwarzweiß-Betrieb schalten diese Befehle nur die Rahmenfarbe weiter, den aktuellen Modus verändern sie nicht.

SHIFT F und SHIFT B sind dagegen in beiden Betriebsarten (farbig und schwarzweiß) gleich: Sie färben den gesamten Vorder- oder Hintergrund mit der aktuellen Rahmenfarbe ein. Sollte man auf diese Art im Farb-Betrieb versehentlich eine mühsam erstellte Colorierung löschen: Pfeil-nach-links-Taste drücken, und sie ist wieder da.

SHIFT CLR Bildschirm löschen

Die Farbinformation wird nicht gelöscht und kann somit isoliert werden.

+ Schnelle Cursorgeschwindigkeit (beschleunigend)
— Langsame Cursorgeschwindigkeit

HI-EDDI besitzt einen beschleunigenden Cursor: Er ist langsam genug, um durch Antippen des Joysticks pixelweise zu rangieren, bei größeren Entfernungen wird er jedoch — ohne lästiges Umschalten — schneller. Da jedoch die Beschleunigung manchmal unerwünscht ist, läßt sie sich durch Drücken der Minus-Taste abschalten.

F1 bis F8: Tabulatoren

Die vier Funktionstasten dienen als Speicher für vier Cursorpositionen: Durch gleichzeitiges Drücken der SHIFT-Taste und einer Funktionstaste wird die momentane Cursorposition gespeichert, durch Drücken einer Funktionstaste allein springt der Cursor wieder genau an die gespeicherte Stelle. F7 wird von den Befehlen L, R und C automatisch belegt.

H — Horizontale Schrittweite
V — Vertikale Schrittweite
F1 bis F8: Schrittweiten speichern

Noch eine Speicherfunktion haben die Funktionstasten: Normalerweise bewegt sich der Cursor in 1-Pixel-Schritten bei der Joysticksteuerung, beziehungsweise in 8-Pixel-Schritten bei Steuerung mittels Cursortasten. Diese Schrittweiten sind jedoch — getrennt für horizontale und vertikale Bewegung — frei programmierbar, vier Schrittweitenpaare können auf den Funktionstasten gespeichert werden. Das Anwählen einer Schrittweite erfolgt durch gleichzeitiges Drücken der C = (Commodore-)Taste und einer Funktionstaste (gilt nur für die Cursortastensteuerung, der Joystick holt seine Schrittweite immer aus F1), das Programmieren durch die Befehle H und V, wonach die aktuelle (angewählte) Schrittweite angezeigt wird und geändert werden kann.

Im Einschaltzustand sind die Tasten folgendermaßen belegt:
F1: H = 1, V = 1, das ist die normale Joysticksteuerung. Durch Vergrößerung auf 2 oder 3 lassen sich im Draw-Modus punktierte Linien zeichnen.
F3 (Diese Taste ist im Einschaltzustand angewählt): H = 8, V = 8, für Textmodus. Vergrößerung ergibt eine gedehnte Schrift oder einen größeren Zeilenabstand.
F5: H = 24, V = 21, = Spritemaße, zur flächendeckenden Bearbeitung mittels Sprite-Editor.
F7: H = 160, V = 96, zur Einteilung des Bildschirmes in Viertelbilder für Walk-Befehl.

Weitere Anwendungsmöglichkeiten sind: Maßstäbe, Gitterraster, exakt symmetrische Zeichnungen etc...

Space: Sprite-Editor

Durch Drücken der Leertaste kommt man in den Sprite-Editor. Das Setzen und Löschen von Punkten geht dort genauso wie im Draw-Modus. Die Befehle des Grafik-Editors sind im Sprite-Editor nicht zugänglich, dafür stehen die folgenden zur Verfügung:

M Mirror — Das Sprite wird zur Senkrechten gespiegelt.

T Turn — Drehung um 180 Grad. M und T ergeben eine Spiegelung zur Waagrechten.

R Rotate — Das Sprite wird um 90 Grad im Uhrzeigersinn gedreht. Da es jedoch 24 Punkte breit, aber nur 21 hoch ist, gehen die rechten drei Spalten verloren. Außerdem ist zweimal R nicht dasselbe wie T.

G Grid — Zum besseren Abzählen von Punkten wird ein Gitter eingeblendet, bei nochmaliger Eingabe von G wird es wieder ausgeblendet.

SHIFT CLR **Sprite löschen**
Space **Sprite-Editor verlassen**
W Walk **Bildfolge ablaufen lassen**
SHIFT W **Bildsequenz programmieren**

Jetzt lernen die Bilder laufen! Mit W werden die Bildschirmspeicher in schneller, programmierbarer Folge zyklisch durchgeschaltet. Da jedoch sechs Bilder für einen Bewegungsablauf recht wenig sind (Speicher 7 wird als »Leinwand« benutzt, sein Inhalt geht verloren) und außerdem die Bildfolge-Geschwindigkeit bei der Verschiebung von jedesmal 8 KByte nicht gerade hoch ist (maximal 7 Bilder/s), können die sechs Bildschirme in 24 Viertelbilder (mit je 160 x 96 Punkten, die unterste Zeile bleibt frei) zerlegt werden, die bei Maximalgeschwindigkeit in einer Sekunde »durchgerasselt« werden.

Bei der Erstellung solcher Viertelbilder ist die programmierbare Schrittweite (160/96) sowie der Move-Befehl besonders nützlich. Die Programmierung der Bildfolge geschieht durch einen Sequenz-String, der aus den Zahlen 1 bis 6 zum Aufruf der großen Bilder oder aus den Buchstaben A bis X zum Aufruf der Viertelbilder bestehen kann. Der Bildschirmspeicher 1 enthält die Viertelbilder A (links oben), B (rechts oben), C (links unten) und D (rechts unten). Das geht so weiter bis zum Speicher 6, der die Viertelbilder U, V, W, X enthält. Beispiel: Zum sequentiellen Durchschalten aller Viertelbilder besteht der Sequenzstring aus den Buchstaben A bis X in alphabetischer Reihenfolge. Zahlen und Buchstaben dürfen mehrfach und sogar gemischt vorkommen.

Mit SHIFT W gelangt man in den Sequenzstring-Editor, mit W wird die Bewegung gestartet. Während des Laufes kann die Geschwindigkeit mit der Plus- und Minus-Taste geregelt werden, bei gedrückter SHIFT-Taste läuft der Film rückwärts. Durch Druck auf den Feuerknopf wird die Vorführung beendet.

Bei Farb-Betrieb wird Bildschirm 6 als Leinwand verwendet, es stehen nur noch 5 große oder 20 kleine Bilder zur Verfügung.

Z Zeichensatz

HI-EDDI kann auch als Zeichensatz-Editor verwendet werden: Mittels Z wird der gerade angewählte Zeichensatz in die ersten sieben Zeilen des Grafikbildschirmes kopiert. Dort kann er mittels Get, Sprite-Editor und Stamp modifiziert, anschließend auf Diskette gespeichert und von anderen Programmen, zum Beispiel Textverarbeitungsprogramme, verwendet werden.

C= L LOAD (C = bedeutet Commodore-Taste)

C= S SAVE

C= D Directory anzeigen

C= C Commando an Disk oder (wenn nur RETURN) Fehlerkanal lesen

Bei LOAD und SAVE hat man die Wahl zwischen (schwarzweißem) Grafikbild, Farbbild, Zeichensatz und Sprite. LOAD und SAVE beziehen sich immer auf den aktuellen (= sichtbaren) Bildschirmspeicher (beziehungsweise Sprite). Bei SAVE Zeichensatz muß dieser in den ersten sieben Zeilen des sichtbaren Bildschirmes stehen. Bei LOAD Sprite sollte man wirklich nur ein Sprite und nichts längeres laden, sonst gibt's einen netten Absturz. Es können auch Diashow-Bilder und Files anderer Grafikprogramme gelesen werden.

C= P Print — Ausdrucken

Die Druckerroutine HI-PRINT wird bei Bedarf in Overlaytechnik nachgeladen, nach Beendigung des Druckvorganges wird wieder HI-EXE geladen.

Die vorliegende Druckerroutine bietet folgende Möglichkeiten:

1. Ausdruck eines Bildes, groß oder klein.
2. Zwei Bilder nahtlos nebeneinander, was natürlich nur in klein geht.

Superhardcopy

Vor und nach dem Ausdruck werden keine zusätzlichen Zeilenvorschübe ausgegeben, aufeinanderfolgende Ausdrucke hängen somit nahtlos aneinander (Bild 1). So läßt sich zum Beispiel mit sechs Bildern eine Superhardcopy mit 640 x 600 Punkten erstellen (Bild 2), lädt man zwischendurch weitere Bilder von Diskette nach, kann man meterlange Bilder ausdrucken.

Leider ist Drucker nicht gleich Drucker! Die vorliegende Druckerroutine (Listing 1) läßt sich nur für Epson-Drucker und kompatible mit Interface verwenden. Deshalb möchte ich im folgenden Kapitel Hinweise zur Erstellung einer Druckerroutine für andere Konfigurationen geben.

Die Speicherbelegung von HI-EDDI

Die Speicherbelegung von HI-EDDI sieht folgendermaßen aus:

\$57-\$60 und \$F9-\$FE	Temporäre Speicherzellen
\$2C0-\$2FE:	Residenter Datenbereich (Sprite-Inhalt)
\$340-\$3FE:	Temporärer Datenbereich (Kreuz-, großes und kleines Rahmen-Sprite
\$801-\$CFF:	Basic-Speicher
\$D00-\$1F4C:	Maschinenprogramm
\$1F80-\$1FFF:	Residenter Datenbereich (Betriebszustände etc.)
\$2000, \$4000, \$6000 etc.:	Grafikspeicher

Die Druckerroutine wird in denselben Bereich wie HI-EXE geladen, also ab \$D00, nach Beendigung des Druckvorganges wird wieder HI-EXE geladen (Listing 1 und 2).

Die vorliegende Druckerroutine enthält auch ein Software-Interface, das den User-Port zur Centronics-Schnittstelle umfunktioniert. Dazu muß ein Kabel mit folgender Pinbelegung verwendet werden:

Pin am User-Port		Pin am Centronics-Stecker	
M	(PA2)	1	(Strobe)
C	(PB0)	2	(DATA 1)
D	(PB1)	3	(DATA 2)
E	(PB2)	4	(DATA 3)
F	(PB3)	5	(DATA 4)
H	(PB4)	6	(DATA 5)
J	(PB5)	7	(DATA 6)
K	(PB6)	8	(DATA 7)
L	(PB7)	9	(DATA 8)
B	(FLAG2)	10 oder 11	(ACKNLG oder BUSY)
A	(GND)	16	

Will man eine Routine für andere Drucker schreiben, so darf diese den gesamten Speicherbereich von \$D00 bis \$1F7F belegen, lediglich die residenten Speicherbereiche und der Basic-Speicher sind tabu. Die Information darüber, wo welcher

Bildschirm im Speicher liegt, steht in den 7 Bytes von $1F96 bis $1F9C: In $1F96 steht die Nummer des Bildes, das ab $2000 in Speicher liegt (das ist das sichtbare Bild!), in Zelle $1F97 steht die Nummer des Bildes ab $4000 und so weiter. Beispiel: Soll Bild Nummer 2 ausgedruckt werden, so sucht man in der Tabelle nach der 2. Steht sie zum Beispiel in Zelle $1F9A, so liegt Bild 2 im Bereich ab $A000 (unter dem Basic-ROM). Im Farb-Betrieb steht in Zelle $1F97 eine 0, weil im Bereich $4000 bis $6000 dann die Farbinformationen für die sechs Bilder liegen und dieser Bereich für Grafikbilder somit gesperrt sein muß.

Für den nicht Maschinensprache-erfahrenen Leser sind dies sicherlich alles spanische Dörfer, aber vielleicht sind einige »Maschinen-Profis« so nett und schicken Lösungen für andere Drucker ein.

HI-EDDI als Menüprogramm

Wem die Eingabe der Befehle über die Tastatur nicht gefällt, der kann HI-EDDI zu einem Menüprogramm nach dem Vorbild des Koala Painter umfunktionieren, bei dem die Befehle über eine bunte, illustrierte Menütafel eingegeben werden. Ehrlich gesagt: Ich finde die Eingabe über Menütafel alles andere als benutzerfreundlich: Menü holen, Cursor auf das gewünschte Feld bugsieren und dann Knopf drücken ist eine wesentlich umständlichere Prozedur als ein einfacher Tastendruck. Dafür macht das Entwerfen und Realisieren einer Menütafel so viel Spaß, daß ich mich entschlossen habe, diese Möglichkeit vorzusehen.

Doch zunächst zurück zu dem Bytewert, der bei Programmstart auf die Frage »Betriebsart« eingegeben werden muß. Er setzt sich wie folgt zusammen:

```
Bit 7:   =0 : Schwarzweiß-Betrieb
         =1 : Farb-Betrieb
Bit 6 :  =0 : nur Tastatureingabe
         =1 : Menü-Eingabe
Bit 5-1 :     keine Funktion
Bit 0 :  =0 : Beim Start werden alle Bildschirmspeicher gelöscht
         =1 : Es wird nichts gelöscht, im Speicher befindliche Bilder
              bleiben erhalten
```

Für Farbe und Menü zum Beispiel muß 192 eingegeben werden. Ich finde die Menü-Eingabe nur in Farbe sinnvoll, ein Schwarzweiß-Menü sieht recht langweilig aus. Ist das Bit 6 gesetzt, also die Menü-Betriebsart angewählt, so muß sich auf der Diskette unter dem Namen »MENUE« die Menütafel befinden, die beim Start automatisch nachgeladen wird. Das Erstellen einer solchen Menütafel soll nun erklärt werden:

Die Menütafel wird in 20 x 12, also insgesamt 240 Felder von je 16 x 16 Punkten eingeteilt. 20 x 16 = 320, die Breite des Bildschirms, aber 12 x 16 = 192 gibt nicht ganz die Höhe des Bildschirmes: Die letzte Zeile (acht Punkte) bleibt, wie bei Walk, frei. In dieser Zeile werden die Steuerbytes untergebracht, für jedes der 240 Felder 2 Bytes. Wird im Menü-Modus der Cursor auf eines der Felder gefahren und der Feuerknopf gedrückt, dann holt sich HI-EDDI aus der letzten Zeile die diesem Feld zugeordneten Steuerbytes und entnimmt aus ihnen, welcher Befehl ausgeführt werden soll. Die Zuordnung von Feldern zu Steuerbytes geht zeilenweise, der Reihe nach: Die ersten beiden Steuerbytes (Adresse $3E00 und $3E01, wenn die Menütafel angewählt ist) gehören zum ersten Feld links oben, die nächsten beiden Bytes zu dem daneben und so weiter. Die Steuerbytes zum letzten Feld haben die Adressen $3FDE und $3FDF, das liegt bereits im unsichtbaren Teil des Bildschirmspeichers. Die nicht gerade ansehnlichen Steuerbytes im sichtbaren Teil können durch gleiche Farben für Vorder- und Hintergrund versteckt werden.

Somit ist klar, wie man vorgehen muß: Zunächst ist ein Menü-Bild zu zeichnen, bei dessen Einteilung nur die 20 x 12-Feld-Rasterung zu berücksichtigen ist. Wie man die Befehle anordnet, wieviele Fehler man für einen Befehl zusammenhängt und welche Befehle man überhaupt ins Menü aufnimmt, bleibt jedem selbst überlassen. Denn keineswegs alle Befehle sind »menüfähig«: Die Korrekturtaste (Pfeil nach rechts) wirkt nur bis zum letzten Bildschirmwechsel zurück. Da jedoch das Anwählen der Menütafel ein Bildschirmwechsel ist, verliert die Korrekturtaste dabei ihre Wirkung. Dasselbe gilt für die Pfeil-nach-oben-Taste: Die Anwahl der Menütafel gilt als Moduswechsel! Aus demselben Grund muß beim MOVEn über zwei Seiten die Seitennummer über die Tastatur eingegeben werden, da bei einem Moduswechsel die Markierung eines Quellbereiches gelöscht wird.

Aus dem Textmodus kommt man nicht direkt ins Menü (SPACE wird als zu druckendes Zeichen interpretiert). Es muß daher zuerst ein anderer Modus über die Tastatur angewählt werden (zum Beispiel CTRL D für Draw), bevor das Menü wieder zugänglich ist.

Dafür bietet die Menütafel auch einige Vorteile: Die Farbenwahl erfolgt nicht mehr durch umständliches Fortschalten, sondern durch Direktanwahl der Farben. Und die Befehle U, O, X, die normalerweise zwei Eingaben erfordern, werden jetzt mit einer Eingabe »erschlagen«.

Hat man nun ein Menübild gemalt (und abgespeichert!), so folgt der zweite Teil, das Erstellen der Steuerbytes, am besten in Form eines DATA-Laders (siehe Listing 4). Von den zwei Steuerbytes pro Feld enthält das zweite den Tastaturcode des jeweiligen Befehls. Die Tastaturcodes sind im 64'er, Ausgabe 5/84, Seite 105 zusammengestellt, sie lassen sich auch mit dem folgenden »Programm« ermitteln:

```
10 PRINT PEEK(203):GOTO10
```

Beim ersten Steuerbyte muß man zwischen verschiedenen Befehlskategorien unterscheiden:

1. »Normale« Befehle: Bit 0 : SHIFT-Bit, muß gesetzt sein, wenn für den entsprechenden Befehl die SHIFT-Taste gedrückt werden muß. Bit 1 : C=-Bit, muß bei gedrückter Commodore-Taste gesetzt sein. Die Bits 2 bis 7 müssen 0 sein. Beispiel: Sprite-Editor einschalten ergibt die Bytes 1 (SHIFT, im Menü-Betrieb dient Space zur Anwahl der Menütafel, der Sprite-Editor ist nur mehr über SHIFT-Space erreichbar) und 60 (Tastaturcode Space).
2. Befehle U, O, X: Tastaturcode der entsprechenden Nummerntaste. Beispiel: X 5 ergibt die Steuerbytes 16 (Tastaturcode »5« und 23 (Tastaturcode »X«).
3. Farbbefehle: Bit 0 : SHIFT-Bit. Bits 2 bis 5: Nummer der Farbe, wie im Commodore-Handbuch angegeben. Die Bits 1 und 6 müssen 0 sein, das Bit 7 ist 1. Beispiel: Total Foreground, Blau ergibt: 128 (Bit 7) + 4 x 6 (blau) + 1 (SHIFT) = 153 und 21 (»F«).
4. »Leerer« Befehl ohne Verlassen der Menütafel (zum Beispiel für Überschrift): 1. Byte = 64, 2. Byte = 0
5. »Leerer« Befehl mit Verlassen des Menüs: 1. Byte = 0, 2. Byte = 64.

Die 5 Bytes im Anschluß an die 480 Steuerbytes müssen die Kennung »CBM80« enthalten. Das ist eigentlich die ROM-Kennung, ich habe sie hier als Menütafel-Kennung mißbraucht, um versehentliche Menü-Zugriffe auf ein normales Bild zu vermeiden. Vorsicht: Fehler in den Steuerbytes können zum Programmabsturz führen!

Nun braucht man nur noch Menübild und Steuerbytes zu verknubbeln: Dazu HI-EDDI laden, Starten (Betriebsart 128), Menübild laden und Programm mit STOP/RESTORE unterbrechen. Steuerbyte-DATA-Lader laden und starten. Der Lader muß, wie im Beispiel (Listing 4) zu sehen,mit den Befehlen POKE56, 32:CLR beginnen, um das im Speicher befindliche

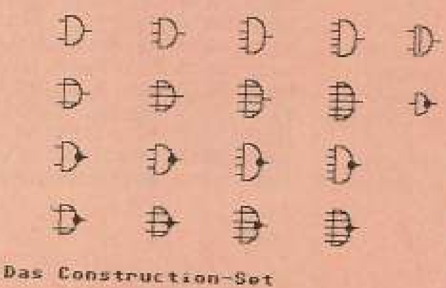
Menübild zu schützen! Dann HI-EDDI laden, starten (Betriebsart 129, damit die jetzt bereits fertige Menütafel im Speicher nicht gelöscht wird) und die nun sichtbare Menütafel unter dem Namen »MENUE« speichern.

Wie wird der »Menü-HI-EDDI« nun bedient? Beim Start muß auf die Frage Betriebsart 192 (oder 193) eingegeben werden. Auf der Menütafel fährt man den Cursor auf den gewünschten Befehl und drückt den Feuerknopf. Daraufhin wird das Menü verlassen, auf den aktuellen Bildschirm zurückgeschaltet (im Einschaltzustand ist das jetzt 2, da in Speicher 1 die Menütafel steckt) und der Befehl ausgeführt. Das Menü wird durch Drücken der Space-Taste auf den Bildschirm geholt. Die Tastatureingabe bleibt neben dem Menü voll erhalten. Wie schon erwähnt, belegt das Menü den Speicher 1. Dieser Speicher kann natürlich auch direkt angewählt werden, allerdings befindet man sich dann nicht im Menü-Modus, sondern die Menütafel wird wie ein ganz normales Bild behandelt!

Zum Abschluß noch ein Tip: HI-EDDI läßt sich mit STOP/RESTORE unterbrechen (nicht während einer Befehlsausführung!), ein Warmstart ist mit GOTO150 möglich, ein Kaltstart ohne Nachladen von HI-EXE, aber mit Betriebsart-Eingabe durch RUN100.

(Hans Haberl/gk)

4-Bit-ALU (Arithmetic logic unit) Typ 74181
(Beispiel für einen Ausdruck mit 640x600 Punkten)



Das Construction-Set

Bild 2. Dieser verkleinerte Schaltplan wurde aus insgesamt sechs verschiedenen Bildern zusammengesetzt. Je zwei nebeneinander und drei untereinander. Auch ein reverser Ausdruck ist mit einem Tastendruck möglich. Die Symbole aus dem Construction-Set können einzeln in jedes beliebige Bild hineinkopiert werden.

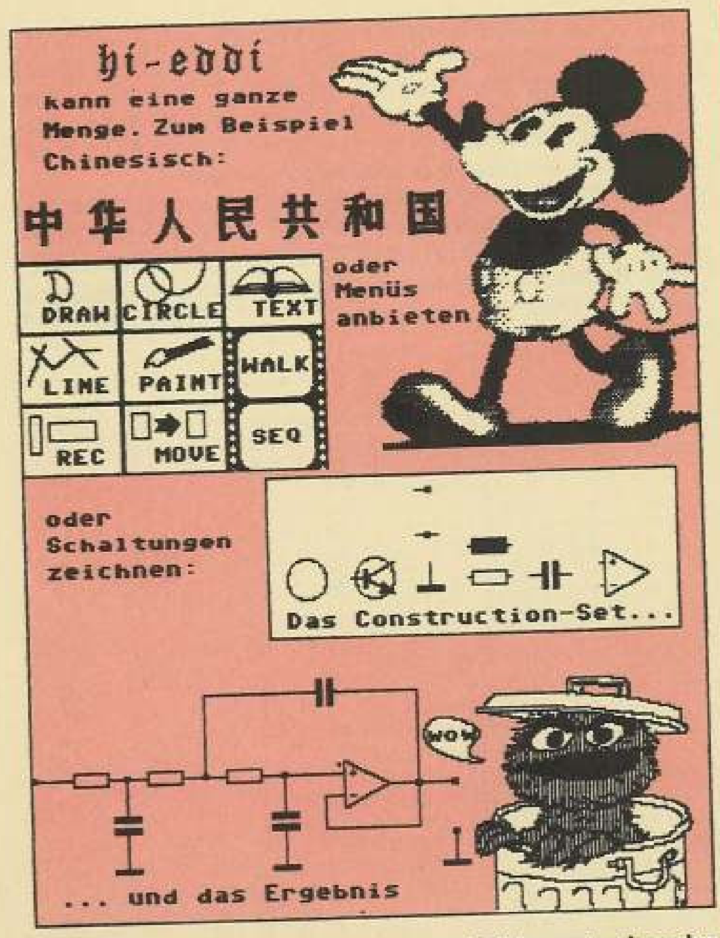


Bild 1. In dieser Hardcopy wurden zwei Bilder untereinander ausgedruckt. Dies geschieht nahtlos. Es können zwei Bilder nebeneinader und unbegrenzt viele untereinander gesetzt werden. Siehe dazu auch Bild 2.

Listing 2. HI-EXE ist das umfangreichste Unterprogramm im HI-EDDI-System. Nach dem Starten speichert es sich selbst als Maschinenprogramm auf Diskette. Lesen Sie vor dem Abtippen den Beitrag »Checksummer« in dieser Ausgabe.

```
                                                    <208>
1 REM ***********************************          <133>
2 REM *              HI-EXE                 *      <230>
3 REM *                                     *      <094>
4 REM *        PROGRAMM WIRD AUF            *      <232>
5 REM *         FLOPPY ERZEUGT              *      <233>
6 REM *                                     *      <214>
7 REM ***********************************          <066>
8 :                                                <037>
9 OPEN 2,8,2,"@:HI-EXE,P,W"                        <058>
10 PRINT#2,CHR$(0);CHR$(13);                       <093>
15 DIM H(75) : FOR I=0 TO 9                        <250>
20 H(48+I)=I : H(65+I)=I+10 : NEXT
30 FOR I=3328 TO 8013 : READ A$
   : REM HIER AKTUELLE WERTE EINSETZEN !           <165>
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1))          <063>
50 D=H(H)*16+H(L) : S=S+D :PRINT#2,CHR$(D);
                        <242>
60 A=A+1:IF A<9  THEN NEXT : A=-1                  <111>
65 PRINT "ZEILE:";1000+Z;                          <012>
70 READ V : Z=Z+1 : IF V=S THEN 85                 <210>
80 PRINT" PRUEFSUMMENFEHLER !";999+Z:CLOSE 2
                                                   <027>
   :STOP                                           <055>
85 IF A<0 THEN CLOSE 2: END                        <065>
90 S=0 : A=0 : PRINT : NEXT : CLOSE 2:END          <153>
95 :                                               <154>
96 :                                               <155>
97 :                                               <156>
98 :                                               <157>
99 :                                               <155>
1000 DATA 4C,11,1B,4C,C1,1E,4C,36,1F, 580  <127>
1001 DATA 20,74,1C,20,24,0D,AD,83,1F, 592  <088>
1002 DATA 20,D3,1C,20,67,12,20,D1,13, 684  <248>
1003 DATA EE,28,D0,EE,29,D0,4C,0C,0D, 1074 <239>
1004 DATA A5,CB,CD,80,1F,F0,28,8D,80, 1281 <212>
1005 DATA 1F,C9,40,F0,21,A8,AD,8D,02, 1053 <183>
1006 DATA 8D,81,1F,AA,29,04,D0,0A,AD, 907  <119>
1007 DATA 84,1F,C9,01,D0,03,4C,12,10, 686  <250>
1008 DATA 98,A2,2A,DD,6E,0D,F0,04,CA, 1146 <219>
1009 DATA 10,F8,60,8A,AE,84,1F,E0,0C, 1071 <124>
1010 DATA D0,05,48,20,0D,11,68,0A,AA, 631  <219>
1011 DATA BD,9A,0D,48,BD,99,0D,48,8A, 993  <098>
1012 DATA 4A,60,33,38,3B,08,0B,10,13, 390  <095>
1013 DATA 18,1D,1F,1C,15,04,05,06,03, 151  <087>
1014 DATA 12,16,2A,11,14,29,24,09,0E, 219  <120>
1015 DATA 1A,0D,0A,21,26,17,1E,0C,20, 217  <100>
1016 DATA 23,02,07,00,3C,39,36,28,2B, 298  <138>
1017 DATA 54,0E,63,0E,63,0E,63,0E,63, 536  <166>
1018 DATA 0E,63,0E,63,0E,63,0E,F7,0D, 613  <069>
1019 DATA F7,0D,ED,0E,ED,0E,BA,0F,BA, 1149 <245>
1020 DATA 0F,BA,0F,BA,0F,EE,0D,06,0E, 688  <241>
1021 DATA EE,0D,06,0E,EE,0D,EE,0D,06, 779  <159>
1022 DATA 0E,23,11,06,0E,06,0E,EE,0D, 357  <226>
1023 DATA 06,0E,3D,0E,BF,0E,C5,0E,B9, 696  <100>
1024 DATA 0E,B2,10,00,10,04,10,BC,12, 450  <170>
1025 DATA 3A,13,FA,12,E2,10,73,1C,F4, 974  <029>
1026 DATA 18,F6,0F,FA,0F,AA,AD,81,1F, 1053 <171>
1027 DATA 29,02,F0,0F,8A,29,0F,8D,80, 761  <151>
1028 DATA 1F,20,47,1C,20,E1,1C,68,68, 655  <224>
1029 DATA 60,8A,29,0F,8D,84,1F,AA,BD, 953  <205>
1030 DATA 24,0E,8D,F9,07,BD,31,0E,8D, 840  <255>
1031 DATA 15,D0,A9,00,8D,AA,1F,8D,A9, 1050 <206>
1032 DATA 1F,4C,5F,13,0D,0E,0D,0D,0D, 287  <188>
1033 DATA 0D,0D,0D,0F,0F,0F,0F,0D,02, 114  <248>
1034 DATA 02,02,02,02,02,02,02,02,02, 18   <104>
1035 DATA 03,03,02,A2,20,86,58,A0,00, 584  <234>
1036 DATA 84,57,B1,57,49,FF,91,57,C8, 1243 <040>
1037 DATA D0,F7,E6,58,CA,D0,F2,60,AD, 1694 <231>
1038 DATA 81,1F,29,01,F0,F8,A9,51,8D, 1081 <236>
1039 DATA A9,1F,AD,96,1F,20,CC,0E,48, 876  <253>
1040 DATA 86,FC,78,A9,34,85,01,AD,A9, 1203 <239>
1041 DATA 1F,D0,29,B1,57,AA,B1,59,91, 1125 <001>
1042 DATA 57,8A,91,59,C8,D0,F3,E6,58, 1428 <031>
1043 DATA E6,5A,C6,FC,D0,EB,68,20,80, 1477 <041>
1044 DATA 0F,A8,AE,96,1F,B9,96,1F,8D, 1045 <255>
1045 DATA 96,1F,8A,99,96,1F,D0,17,8D, 1025 <213>
1046 DATA A2,0E,B1,59,31,57,91,57,C8, 1010 <049>
1047 DATA D0,F7,E6,58,E6,5A,CA,D0,F0, 1743 <201>
1048 DATA 8E,A9,1F,68,A9,37,85,01,58, 892  <194>
1049 DATA 60,A9,31,8D,A9,1F,60,A9,11, 937  <024>
1050 DATA 8D,A9,1F,60,A9,51,8D,A9,1F, 1028 <228>
1051 DATA 60,A0,06,D9,96,1F,F0,03,88, 1039 <212>
1052 DATA D0,F8,98,48,18,A9,00,69,20, 1010 <227>
1053 DATA 88,10,FB,85,5A,A2,20,86,58, 1042 <144>
1054 DATA A0,00,84,57,84,59,68,60,29, 841  <029>
1055 DATA 01,85,FD,AD,89,1F,29,FE,05, 1028 <027>
1056 DATA FD,8D,89,1F,AA,AD,81,1F,29, 1106 <023>
1057 DATA 01,D0,0F,EE,20,D0,EE,27,D0, 1187
```

```
1058 DATA BA,10,05,A9,07,20,07,0E,60, 484  <153>
1059 DATA AD,F9,07,48,A2,04,86,58,A0, 1049 <254>
1060 DATA 00,84,57,18,20,5F,1A,C8,D0, 804  <178>
1061 DATA F9,E6,58,CA,D0,F4,F0,3E,AD, 1696 <090>
1062 DATA F9,07,48,AD,88,1F,A2,04,86, 968  <227>
1063 DATA 58,A0,00,84,57,91,57,C8,D0, 1107 <222>
1064 DATA F8,E6,58,CA,D0,F6,F0,23,AE, 1674 <082>
1065 DATA 89,1F,10,E0,A9,40,85,58,A2, 1024 <236>
1066 DATA 04,86,5A,D0,0D,AE,89,1F,10, 807  <219>
1067 DATA 1D,A9,40,85,5A,A2,04,86,58, 873  <200>
1068 DATA AD,F9,07,48,20,C9,10,68,8D, 995  <230>
1069 DATA F9,07,A9,08,8D,F8,07,A9,0D, 1014 <039>
1070 DATA 8D,FA,07,AE,00,04,8E,88,1F, 895  <254>
1071 DATA 60,48,AD,84,1F,C9,06,D0,0A, 929  <231>
1072 DATA AD,AA,1F,C9,02,D0,03,20,48, 892  <231>
1073 DATA 0F,AD,89,1F,10,18,AD,96,1F, 750  <248>
1074 DATA 20,B1,0F,20,5E,0F,68,48,A8, 709  <213>
1075 DATA B9,96,1F,20,B1,0F,20,4F,0F, 716  <226>
1076 DATA 20,57,0F,68,60,A8,A9,40,18, 759  <196>
1077 DATA 69,04,88,D0,FB,60,29,03,AA, 1014 <003>
1078 DATA AD,B1,1F,29,03,C9,01,F0,19, 844  <217>
1079 DATA 29,02,D0,28,BD,9D,1F,8D,85, 942  <247>
1080 DATA 1F,BD,A1,1F,8D,86,1F,BD,A5, 1072 <088>
1081 DATA 1F,8D,87,1F,4C,5F,13,AD,85, 834  <012>
1082 DATA 1F,9D,9D,1F,AD,86,1F,9D,A1, 1032 <084>
1083 DATA 1F,AD,87,1F,9D,A5,1F,60,8E, 961  <029>
1084 DATA BA,1F,60,A9,06,D0,02,A9,0C, 831  <228>
1085 DATA 8D,94,1F,60,A2,80,D0,02,A2, 1078 <010>
1086 DATA 00,AD,81,1F,29,04,F0,03,8E, 763  <219>
1087 DATA 95,1F,60,E0,03,F0,FB,8A,0A, 1142 <037>
1088 DATA AA,BD,79,EB,85,57,BD,7A,EB, 1481 <126>
1089 DATA 85,58,B1,57,AA,29,7F,C9,20, 1056 <033>
1090 DATA B0,06,AC,80,1F,4C,48,0D,8A, 812  <251>
1091 DATA 29,E0,C9,60,D0,04,A9,40,D0, 1215 <011>
1092 DATA 09,8A,10,04,09,40,D0,02,29, 491  <170>
1093 DATA BF,29,7F,85,FD,8A,29,1F,05, 960  <030>
1094 DATA FD,0D,95,1F,A2,00,86,58,0A, 840  <247>
1095 DATA 26,58,0A,26,58,0A,26,58,85, 531  <193>
1096 DATA 57,AD,18,D0,29,02,0A,0A,09, 564  <224>
1097 DATA D0,05,58,85,58,78,A9,33,85, 995  <223>
1098 DATA 01,A0,07,B1,57,99,BA,1F,88, 938  <245>
1099 DATA 10,F8,A9,37,85,01,58,20,11, 759  <197>
1100 DATA 15,A2,00,20,BE,14,A9,08,85, 735  <212>
1101 DATA 5F,A0,00,3E,BA,1F,A9,00,90, 847  <004>
1102 DATA 02,A5,5E,11,57,91,57,20,06, 635  <189>
1103 DATA 15,C6,5F,D0,EC,E6,58,E8,E0, 1535 <106>
1104 DATA 08,D0,DC,A9,00,8D,81,1F,4C, 982  <022>
1105 DATA BD,12,AD,18,D0,29,02,0A,0A, 675  <000>
1106 DATA 09,D0,85,58,A9,20,85,5A,A2, 1024 <021>
1107 DATA 08,78,A9,33,85,01,A0,00,84, 774  <202>
1108 DATA 57,84,59,B1,57,91,59,C8,D0, 1214 <027>
1109 DATA F9,E6,58,E6,5A,CA,D0,F2,A9, 1708 <122>
1110 DATA 37,85,01,58,60,AD,89,1F,49, 787  <244>
1111 DATA 40,29,40,0D,81,1F,F0,03,4C, 661  <222>
1112 DATA 27,1D,A0,03,B9,84,1F,99,AF, 907  <020>
1113 DATA 1F,88,10,F7,AD,96,1F,8D,AE, 1099 <113>
1114 DATA 1F,A9,01,20,64,0E,A9,0C,4C, 604  <254>
1115 DATA 07,0E,AD,AE,1F,20,64,0E,A0, 705  <026>
1116 DATA 03,B9,AF,1F,99,84,1F,88,10, 862  <017>
1117 DATA F7,AD,84,1F,4C,07,0E,AD,81, 982  <057>
1118 DATA 1F,29,01,F0,03,4C,F8,0D,AD, 826  <027>
1119 DATA 15,D0,48,A2,00,8E,15,D0,8E, 976  <005>
1120 DATA A9,1F,86,FC,E8,86,FD,AD,89, 1515 <146>
1121 DATA 1F,2A,2A,29,01,49,07,20,64, 369  <223>
1122 DATA 0E,A0,04,B1,2D,99,59,00,88, 778  <254>
1123 DATA D0,F8,C6,5B,10,01,60,A4,FC, 1274 <076>
1124 DATA B1,5C,AE,8D,02,F0,0A,C6,FC, 1286 <128>
1125 DATA 10,10,A6,5B,86,FC,D0,0A,E6, 1123 <070>
1126 DATA FC,C4,5B,90,04,A2,00,86,FC, 1235 <082>
1127 DATA C9,40,B0,2C,29,07,8D,AE,1F, 879  <052>
1128 DATA 20,CC,0E,78,A9,34,85,01,B1, 902  <004>
1129 DATA 59,91,57,C8,D0,F9,E6,58,E6, 1526 <095>
1130 DATA 5A,CA,D0,F2,AD,89,1F,10,09, 1108 <100>
1131 DATA AD,AE,1F,20,B1,0F,20,4F,0F, 728  <062>
1132 DATA 4C,3D,12,38,E9,01,48,4A,4A, 665  <017>
1133 DATA 29,07,18,69,01,8D,AE,1F,20, 556  <006>
1134 DATA CC,0E,68,48,4A,90,04,A2,A0, 938  <032>
1135 DATA 86,59,4A,90,06,A5,5A,09,0F, 726  <012>
1136 DATA 85,5A,A9,27,85,58,A9,D0,85, 1162 <072>
1137 DATA 57,78,A9,34,85,01,A2,0C,A0, 896  <012>
1138 DATA A0,88,B1,59,91,57,98,D0,F8, 1402 <071>
1139 DATA A5,57,18,69,40,85,57,A5,58, 918  <002>
1140 DATA 69,01,85,58,A5,59,18,69,40, 774  <243>
1141 DATA 85,59,A5,5A,69,01,85,5A,CA, 1008 <079>
1142 DATA D0,D9,68,AA,AD,89,1F,10,36, 1110 <100>
1143 DATA AD,AE,1F,20,B1,0F,85,58,A9, 992  <079>
1144 DATA 00,85,57,8A,4A,90,04,A2,14, 762  <246>
1145 DATA 86,57,4A,90,0D,A5,57,18,69, 833  <016>
1146 DATA E0,85,57,A5,58,69,01,85,58, 1024 <047>
1147 DATA A9,FA,85,59,A9,04,85,5A,A2, 1199 <114>
1148 DATA 0B,A0,13,8C,C5,1F,20,1F,1A, 647  <044>
1149 DATA A9,37,85,01,58,A5,FD,20,D3, 1107 <081>
1150 DATA 1C,A5,CB,C9,28,D0,03,38,26, 945  <060>
1151 DATA FD,C9,28,D0,02,46,FD,AD,00, 1200 <116>
1152 DATA DC,29,10,F0,03,4C,5A,11,68, 807  <027>
1153 DATA 8D,15,D0,4C,E8,1C,20,A2,12, 918  <044>
1154 DATA 29,0F,D0,06,A9,28,8D,83,1F, 782  <051>
1155 DATA 60,48,29,03,F0,0A,29,01,8D, 645  <004>
1156 DATA 81,1F,A2,00,20,3E,13,68,29, 580  <254>
1157 DATA 0C,F0,0C,4A,4A,29,01,8D,81, 724  <044>
1158 DATA 1F,A2,00,20,C0,12,AE,83,1F, 771  <034>
1159 DATA EC,94,1F,90,05,CA,CA,8E,83, 1241 <140>
1160 DATA 1F,60,78,AE,02,DC,A0,00,8C, 943  <066>
1161 DATA 02,DC,AD,00,DC,CD,00,DC,D0, 1248 <162>
1162 DATA F8,8E,02,DC,58,49,FF,8D,82, 1299 <162>
1163 DATA 1F,60,AE,8A,1F,AC,86,1F,AD, 980  <128>
1164 DATA 81,1F,29,01,D0,1A,18,98,7D, 737  <039>
1165 DATA 8B,1F,90,24,EE,87,1F,AE,87, 1063 <143>
1166 DATA 1F,E0,02,90,1A,A9,01,8D,87, 873  <050>
1167 DATA 1F,A9,F0,B0,11,38,98,FD,8B, 1233 <134>
1168 DATA 1F,B0,0A,CE,87,1F,10,05,A9, 779  <082>
1169 DATA 00,8D,87,1F,8D,86,1F,4C,5F, 784  <092>
1170 DATA 13,AD,84,1F,C9,01,D0,38,AD, 994  <088>
1171 DATA 81,1F,29,01,49,01,8D,81,1F, 577  <031>
1172 DATA F0,03,20,BD,12,20,11,15,A2, 714  <004>
1173 DATA 00,20,BE,14,A9,08,85,5F,A0, 807  <048>
1174 DATA 00,A5,5E,49,FF,31,57,91,57, 955  <059>
1175 DATA 20,06,15,C6,5F,D0,F1,E6,58, 1122 <111>
1176 DATA E8,E0,08,D0,E1,AD,81,1F,F0, 1470 <150>
1177 DATA 83,60,AE,8A,1F,AC,85,1F,AD, 1079 <177>
1178 DATA 81,1F,29,01,D0,0B,18,98,7D, 722  <047>
1179 DATA 8F,1F,90,0D,A9,FF,B0,09,38, 996  <116>
1180 DATA 98,FD,8F,1F,B0,02,A9,00,8D, 1067 <151>
1181 DATA 85.1F.20.B8.13.BD.C5.13.CD. 1012 <139>
1182 DATA 85,1F,90,03,8D,85,1F,BD,C6, 1003 <132>
1183 DATA 13,CD,85,1F,B0,03,8D,85,1F, 872  <087>
1184 DATA AD,87,1F,D0,0D,BD,C7,13,CD, 1172 <188>
1185 DATA 86,1F,90,10,8D,86,1F,B0,0B, 818  <073>
1186 DATA BD,C8,13,CD,86,1F,B0,03,8D, 1098 <169>
1187 DATA 86,1F,AD,85,1F,8D,01,D0,8D, 993  <118>
1188 DATA 03,D0,AD,86,1F,8D,00,D0,8D, 1039 <145>
1189 DATA 02,D0,AD,87,1F,0A,0D,87,1F, 738  <102>
1190 DATA 85,FD,AD,10,D0,29,FC,05,FD, 1334 <184>
1191 DATA 8D,10,D0,60,AD,F9,07,38,E9, 1179 <151>
1192 DATA 0D,0A,0A,AA,60,28,EF,0E,4D, 669  <128>
1193 DATA 28,EB,11,49,32,E5,18,40,AD, 908  <088>
1194 DATA 82,1F,29,10,F0,10,20,11,15, 544  <008>
1195 DATA AD,84,1F,0A,AA,BD,EA,13,48, 1030 <185>
1196 DATA BD,E9,13,48,60,02,14,E7,13, 881  <065>
1197 DATA 68,15,0D,16,55,16,98,17,74, 561  <042>
1198 DATA 18,42,1A,18,14,86,14,42,14, 400  <002>
1199 DATA 45,14,95,1A,20,BE,14,AC,8D, 819  <096>
1200 DATA 02,F0,09,A0,00,49,FF,31,57, 875  <066>
1201 DATA 91,57,60,11,57,91,57,60,A5, 925  <032>
```

Listing 2. HI-EXE (Fortsetzung)

```
1202 DATA 5B,48,A2,00,20,BE,14,A9,18, 760  <073>
1203 DATA 85,5F,A0,00,A5,5E,49,FF,31, 1024 <147>
1204 DATA 57,91,57,20,06,15,C6,5F,D0, 879  <071>
1205 DATA F1,E6,5B,E8,E8,E8,E0,3F,D0, 1753 <218>
1206 DATA DF,68,85,5B,60,20,19,14,A2, 886  <085>
1207 DATA 00,20,BE,14,A9,18,85,5F,A0, 823  <081>
1208 DATA 00,BD,C2,02,8D,BC,1F,BD,C1, 1127 <197>
1209 DATA 02,8D,BB,1F,BD,C0,02,8D,BA, 1071 <200>
1210 DATA 1F,2E,BC,1F,2E,BB,1F,2E,BA, 792  <197>
1211 DATA 1F,A9,00,90,02,A5,5E,11,57, 709  <072>
1212 DATA 91,57,20,06,15,C6,5F,D0,E6, 1022 <123>
1213 DATA E6,5B,E8,E8,E8,E0,3F,D0,C2, 1706 <222>
1214 DATA 60,A9,03,8D,15,D0,A2,00,20, 832  <060>
1215 DATA BE,14,A9,18,85,5F,A0,00,18, 815  <097>
1216 DATA B1,57,25,5E,F0,01,38,3E,C2, 948  <103>
1217 DATA 02,3E,C1,02,3E,C0,02,20,06, 553  <054>
1218 DATA 15,C6,5F,D0,E8,E6,5B,E8,E8, 1539 <222>
1219 DATA E8,E0,3F,D0,D6,A9,0B,8D,84, 1394 <212>
1220 DATA 1F,60,A9,01,85,58,A5,5B,29, 815  <095>
1221 DATA F8,85,57,0A,26,58,0A,26,58, 740  <086>
1222 DATA 18,65,57,90,02,E6,58,0A,26, 724  <066>
1223 DATA 58,0A,26,58,0A,26,58,85,57, 580  <073>
1224 DATA A5,5B,29,07,85,60,A5,5C,29, 831  <098>
1225 DATA F8,18,65,60,65,57,85,57,A5, 1042 <129>
1226 DATA 5D,65,58,85,58,A5,5C,29,07, 808  <104>
1227 DATA A8,A9,00,38,6A,88,10,FC,85, 1036 <164>
1228 DATA 5E,60,46,5E,90,06,66,5E,98, 852  <104>
1229 DATA 69,08,A8,60,20,BB,13,AD,85, 921  <110>
1230 DATA 1F,38,FD,C5,13,85,5B,AD,86, 1087 <208>
1231 DATA 1F,38,FD,C7,13,85,5C,AD,87, 1091 <208>
1232 DATA 1F,E9,00,85,5D,60,20,E8,1C, 878  <126>
1233 DATA AD,AA,1F,49,01,8D,AA,1F,F0, 1030 <223>
1234 DATA 2D,A2,02,B5,5B,9D,AB,1F,CA, 1042 <224>
1235 DATA 10,F8,A2,03,20,E0,0F,AD,02, 875  <115>
1236 DATA D0,8D,04,D0,AD,03,D0,8D,05, 1091 <183>
1237 DATA D0,AD,10,D0,6A,08,2A,28,2A, 843  <132>
1238 DATA 8D,10,D0,A9,06,8D,15,D0,68, 1014 <164>
1239 DATA 68,60,20,2E,15,20,03,14,A5, 519  <060>
1240 DATA 5B,48,A5,5C,48,A5,5D,48,AD, 995  <175>
1241 DATA AC,1F,38,E5,5C,48,AD,AD,1F, 1029 <253>
1242 DATA E5,5D,8D,BD,1F,B0,0B,68,49, 1047 <228>
1243 DATA FF,69,01,48,A9,00,ED,BD,1F, 1059 <229>
1244 DATA 8D,BB,1F,8D,BF,1F,68,8D,BA, 1153 <024>
1245 DATA 1F,8D,BE,1F,AD,AB,1F,18,E5, 1021 <007>
1246 DATA 5B,90,04,49,FF,69,FE,8D,BC, 1255 <246>
1247 DATA 1F,6E,BD,1F,38,ED,BA,1F,AA, 1041 <023>
1248 DATA A9,FF,ED,BB,1F,85,FC,B0,06, 1446 <026>
1249 DATA 0A,0A,2A,20,74,17,AD,BE,1F, 627  <160>
1250 DATA 6D,BC,1F,8D,BE,1F,AD,BF,1F, 1085 <055>
1251 DATA E9,00,4C,F5,15,AD,BD,1F,B0, 1144 <234>
1252 DATA E4,0A,2A,0A,49,02,20,5C,17, 512  <110>
1253 DATA 18,AD,BE,1F,6D,BA,1F,8D,BE, 1075 <040>
1254 DATA 1F,AD,BF,1F,6D,BB,1F,8D,BF, 1085 <059>
1255 DATA 1F,08,20,03,14,28,E8,D0,D8, 790  <122>
1256 DATA E6,FC,D0,D4,68,85,5D,68,85, 1469 <238>
1257 DATA 5C,68,85,5B,60,20,2E,15,AD, 788  <151>
1258 DATA AB,1F,48,A5,5B,8D,AB,1F,20, 905  <198>
1259 DATA 6C,15,68,8D,AB,1F,A5,5C,48, 905  <185>
1260 DATA A5,5D,48,AD,AC,1F,85,5C,AD, 1104 <010>
1261 DATA AD,1F,85,5D,20,6C,15,68,85, 828  <160>
1262 DATA 5D,68,85,5C,A5,5B,48,AD,AB, 1094 <252>
1263 DATA 1F,85,5B,20,6C,15,68,85,5B, 744  <143>
1264 DATA A5,5C,8D,AC,1F,A5,5D,8D,AD, 1173 <036>
1265 DATA 1F,20,6C,15,60,20,2E,15,A2, 549  <118>
1266 DATA 02,B5,5B,9D,C2,1F,CA,10,F8, 1122 <231>
1267 DATA A9,00,20,EA,16,20,03,14,20, 544  <088>
1268 DATA B8,16,85,F9,20,5A,17,08,A9, 910  <144>
1269 DATA 02,20,EA,16,28,2A,0A,45,F9, 700  <134>
1270 DATA 20,5C,17,20,72,17,08,A9,04, 497  <104>
1271 DATA 20,EA,16,28,2A,45,F9,20,74, 836  <140>
1272 DATA 17,38,AD,BC,1F,ED,BE,1F,AD, 1102 <047>
1273 DATA BD,1F,ED,BF,1F,B0,06,20,5A, 983  <229>
1274 DATA 17,4C,A9,16,20,72,17,90,0C, 615  <120>
1275 DATA A2,02,B5,5B,DD,C2,1F,D0,B4, 1270 <249>
1276 DATA CA,10,F6,60,38,A5,5C,ED,AC, 1282 <009>
1277 DATA 1F,A8,A5,5D,ED,AD,1F,08,26, 944  <229>
1278 DATA FD,28,98,B0,04,49,FF,69,01, 1059 <228>
1279 DATA 8D,C1,1F,38,AD,AB,1F,E5,5B, 1116 <029>
1280 DATA 08,26,FD,28,B0,04,49,FF,69, 952  <182>
1281 DATA 01,8D,C0,1F,A5,FD,49,FF,60, 1207 <005>
1282 DATA 48,20,B8,16,AD,C0,1F,A2,00, 868  <173>
1283 DATA 20,36,17,AD,C1,1F,A2,02,20, 702  <138>
1284 DATA 36,17,68,AA,18,A5,57,65,59, 817  <157>
1285 DATA 9D,BA,1F,A5,58,65,5A,9D,BB, 1162 <031>
1286 DATA 1F,8A,F0,23,38,BD,BA,1F,ED, 1143 <034>
1287 DATA BA,1F,48,BD,BB,1F,ED,BB,1F, 1151 <078>
1288 DATA B0,0C,A8,68,49,FF,69,01,48, 966  <194>
1289 DATA 98,49,FF,69,00,9D,BB,1F,68, 1064 <007>
1290 DATA 9D,BA,1F,60,48,A8,A9,08,85, 1020 <242>
1291 DATA FC,A9,00,95,57,16,57,36,58, 908  <166>
1292 DATA 98,0A,A8,90,0B,68,48,18,75, 802  <159>
1293 DATA 57,95,57,90,02,F6,58,C6,FC, 1253 <232>
1294 DATA D0,E8,68,60,A5,F9,29,02,F0, 1337 <237>
1295 DATA 08,A5,5B,F0,36,C6,5B,38,60, 999  <195>
1296 DATA A5,5B,C9,C7,F0,2C,E6,5B,38, 1317 <027>
1297 DATA 60,A5,F9,29,01,F0,10,A5,5C, 1065 <221>
1298 DATA 05,5D,F0,1C,A5,5C,D0,02,C6, 1031 <240>
1299 DATA 5D,C6,5C,38,60,A5,5D,F0,06, 1039 <250>
1300 DATA A5,5C,C9,3F,F0,08,E6,5C,D0, 1299 <037>
1301 DATA 02,E6,5D,38,60,18,60,20,E8, 861  <157>
1302 DATA 1C,A9,01,20,74,17,A9,00,85, 671  <147>
1303 DATA F9,A5,5B,8D,AB,1F,BA,8E,BD, 1365 <098>
1304 DATA 1F,BA,E0,14,90,6C,A5,FD,48, 1203 <020>
1305 DATA A5,FE,48,A5,5C,48,A5,5D,48, 1150 <022>
1306 DATA A5,5B,85,FD,AD,AB,1F,85,FE, 1404 <082>
1307 DATA 20,72,17,90,46,20,23,18,90, 618  <109>
1308 DATA 41,A5,5B,48,AD,AB,1F,48,20, 872  <220>
1309 DATA 23,18,90,05,20,B1,17,90,F6, 830  <138>
1310 DATA 68,85,FE,68,85,FD,A5,F9,49, 1468 <050>
1311 DATA 01,85,F9,20,72,17,20,23,18, 643  <128>
1312 DATA 90,05,20,B1,17,90,F6,A5,F9, 1185 <200>
1313 DATA 49,01,85,F9,20,72,17,AD,00, 798  <175>
1314 DATA DC,29,10,D0,BC,AE,BD,1F,9A, 1221 <069>
1315 DATA 4C,E8,1C,68,85,5D,68,85,5C, 995  <237>
1316 DATA 68,85,FE,68,85,FD,60,A5,FD, 1495 <060>
1317 DATA 85,5B,A5,FE,8D,AB,1F,20,BE, 1208 <072>
1318 DATA 14,A0,00,31,57,D0,36,A5,5B, 834  <169>
1319 DATA F0,0D,C6,5B,20,BE,14,A0,00, 944  <213>
1320 DATA 31,57,F0,F1,E6,5B,A5,5B,AA, 1364 <040>
1321 DATA 20,BE,14,A0,00,31,57,D0,0E, 760  <178>
1322 DATA A5,5E,11,57,91,57,E6,5B,A5, 1081 <004>
1323 DATA 5B,C9,C8,D0,E9,C6,5B,A5,5B, 1478 <080>
1324 DATA 8D,AB,1F,86,5B,38,60,E6,5B, 1041 <036>
1325 DATA AD,AB,1F,C5,5B,B0,B8,60,20, 1151 <043>
1326 DATA E8,1C,AD,AA,1F,C9,03,D0,05, 1051 <048>
1327 DATA A9,00,8D,AA,1F,0A,AA,A5,5C, 948  <024>
1328 DATA 46,5D,6A,4A,4A,9D,BB,1F,A5, 957  <027>
1329 DATA 5B,4A,4A,4A,9D,BA,1F,EE,AA, 1095 <109>
1330 DATA 1F,BA,F0,56,C9,02,D0,6A,A2, 1174 <032>
1331 DATA 00,BD,BC,1F,38,FD,BA,1F,B0, 1110 <077>
1332 DATA 10,BD,BC,1F,A8,BD,BA,1F,9D, 1155 <109>
1333 DATA BC,1F,98,9D,BA,1F,90,E7,9D, 1277 <096>
1334 DATA C4,1F,E8,E0,02,D0,DF,20,57, 1235 <035>
1335 DATA 0F,A2,00,20,FA,19,BC,C2,1F, 849  <243>
1336 DATA 8D,C3,1F,AE,C4,1F,AC,C5,1F, 1168 <108>
1337 DATA A9,BF,91,57,88,10,F9,A5,57, 1245 <035>
1338 DATA 18,69,28,85,57,90,02,E6,58, 853  <188>
1339 DATA CA,10,E8,AD,96,1F,8D,AE,1F, 1150 <092>
1340 DATA 60,AD,AA,1F,C9,03,D0,F8,CE, 1336 <090>
1341 DATA AA,1F,AD,AE,1F,20,64,0E,4C, 801  <029>
1342 DATA C5,18,CE,AA,1F,60,AD,BE,1F, 1118 <104>
1343 DATA 18,6D,C4,1F,8D,C0,1F,C9,19, 950  <014>
1344 DATA B0,EE,AD,BF,1F,18,6D,C5,1F, 1170 <111>
1345 DATA 8D,C1,1F,C9,28,B0,E0,20,48, 1110 <026>
```

Listing 2. HI-EXE (Fortsetzung)

```
1346 DATA 0F,A2,04,20,FA,19,A5,57,38, 796  <233>
1347 DATA ED,C2,1F,A5,58,ED,C3,1F,A9, 1347 <110>
1348 DATA 00,2A,2A,85,F9,69,04,AA,20, 777  <230>
1349 DATA FA,19,84,59,85,5A,A6,F9,20, 1166 <045>
1350 DATA FA,19,A0,03,06,57,26,58,06, 663  <203>
1351 DATA 59,26,5A,88,D0,F5,AD,AE,1F, 1184 <086>
1352 DATA A0,06,D9,96,1F,F0,03,88,D0, 1151 <027>
1353 DATA F8,A5,58,18,88,30,04,69,20, 850  <204>
1354 DATA 90,F9,85,58,A9,00,A0,08,A6, 1117 <020>
1355 DATA F9,F0,04,A9,FF,A0,F8,84,5C, 1549 <100>
1356 DATA 85,5D,AD,C4,1F,85,FB,78,A9, 1299 <108>
1357 DATA 34,85,01,A2,FF,A0,07,B1,57, 1034 <020>
1358 DATA 91,59,88,10,F9,A0,02,B9,57, 1069 <020>
1359 DATA 00,18,65,5C,99,57,00,B9,58, 730  <207>
1360 DATA 00,65,5D,99,58,00,88,88,F0, 947  <225>
1361 DATA E8,E8,EC,C5,1F,90,DA,E0,27, 1556 <125>
1362 DATA D0,DF,C6,FB,10,D0,A9,37,85, 1461 <087>
1363 DATA 01,58,AD,89,1F,10,2F,A2,04, 659  <246>
1364 DATA 20,FA,19,84,59,85,5A,AD,AE, 1098 <085>
1365 DATA 1F,CD,96,1F,F0,05,20,B1,0F, 886  <025>
1366 DATA F0,02,A9,40,18,6D,C3,1F,38, 890  <252>
1367 DATA E9,04,85,58,AD,C2,1F,85,57, 1076 <063>
1368 DATA AE,C4,1F,20,1F,1A,20,57,0F, 624  <011>
1369 DATA 60,A9,00,85,58,BD,BA,1F,0A, 902  <019>
1370 DATA 0A,18,7D,BA,1F,0A,0A,26,58, 522  <013>
1371 DATA 0A,26,58,7D,BB,1F,90,02,E6, 855  <019>
1372 DATA 58,85,57,A8,A5,58,09,04,85, 875  <240>
1373 DATA 58,60,AC,C5,1F,B1,57,91,59, 1082 <057>
1374 DATA 88,10,F9,A5,57,18,69,28,85, 955  <246>
1375 DATA 57,90,02,E6,58,A5,59,18,69, 934  <238>
1376 DATA 28,85,59,90,02,E6,5A,CA,10, 946  <253>
1377 DATA DD,60,20,BE,14,A0,03,46,58, 880  <252>
1378 DATA 66,57,88,D0,F9,A5,58,49,44, 1176 <056>
1379 DATA 85,5A,A5,57,85,59,A0,00,AD, 1030 <049>
1380 DATA 8D,02,4A,AD,89,1F,29,01,D0, 808  <024>
1381 DATA 15,B0,06,AD,20,D0,4C,70,1A, 830  <253>
1382 DATA B1,59,29,0F,85,FD,B1,57,29, 1013 <066>
1383 DATA F0,4C,91,1A,B0,0A,AD,20,D0, 1086 <081>
1384 DATA 0A,0A,0A,0A,4C,89,1A,B1,59, 545  <025>
1385 DATA 29,F0,85,FD,B1,57,29,0F,05, 992  <030>
1386 DATA FD,91,57,60,20,E8,1C,A0,04, 1037 <055>
1387 DATA B9,10,FD,D9,E0,3F,D0,53,88, 1385 <109>
1388 DATA 10,F5,A9,3E,85,58,A5,58,C9, 1170 <090>
1389 DATA C0,90,02,A9,BF,29,F0,85,FD, 1365 <104>
1390 DATA 4A,4A,65,FD,85,FD,A5,5D,4A, 1220 <129>
1391 DATA A5,5C,6A,4A,4A,4A,18,65,FD, 963  <071>
1392 DATA 0A,90,02,E6,58,85,57,A0,01, 855  <245>
1393 DATA B1,57,AA,88,B1,57,29,40,D0, 1147 <069>
1394 DATA 18,B1,57,30,18,C9,04,B0,08, 752  <252>
1395 DATA 8D,81,1F,8A,A8,4C,48,0D,48, 840  <052>
1396 DATA 8A,A8,20,48,0D,68,A8,4C,48, 843  <037>
1397 DATA 0D,60,A8,29,01,8D,81,1F,08, 628  <012>
1398 DATA 98,4A,4A,A8,28,D0,01,88,8C, 993  <045>
1399 DATA 20,D0,8C,27,D0,8A,A8,4C,48, 1081 <088>
1400 DATA 0D,48,29,01,D0,1F,A9,20,85, 700  <003>
1401 DATA 58,A2,E0,78,A9,34,85,01,A9, 1118 <069>
1402 DATA 00,85,57,A8,91,57,C8,D0,FB, 1279 <096>
1403 DATA E6,58,CA,D0,F6,A9,37,85,01, 1332 <102>
1404 DATA 58,A2,00,8E,17,D0,8E,1D,D0, 1002 <082>
1405 DATA 8E,1C,D0,8E,1B,D0,20,C8,1C, 1015 <114>
1406 DATA A9,04,8D,20,D0,A0,25,B9,91, 1081 <073>
1407 DATA 1B,99,84,1F,88,10,F7,68,8D, 987  <059>
1408 DATA 89,1F,10,23,A9,00,8D,97,1F, 711  <024>
1409 DATA AD,89,1F,29,01,D0,17,A9,40, 847  <043>
1410 DATA 85,58,AD,88,1F,A2,1C,A0,00, 911  <047>
1411 DATA 84,57,91,57,C8,D0,FB,E6,58, 1428 <115>
1412 DATA CA,D0,F6,AD,89,1F,29,40,F0, 1342 <136>
1413 DATA 08,20,F2,10,A9,02,8D,AE,1F, 815  <047>
1414 DATA 4C,74,1C,00,8C,AE,00,F6,00, 780  <044>
1415 DATA 01,01,08,18,A0,01,08,15,60, 320  <201>
1416 DATA 20,06,00,01,07,02,03,04,05, 60   <128>
1417 DATA 06,8C,8C,8C,8C,AE,AE,AE,AE, 1262 <210>
1418 DATA 00,00,00,00,A9,93,20,D2,FF, 813  <004>
1419 DATA A9,01,8D,15,D0,8D,10,D0,A9, 1074 <103>
1420 DATA 10,8D,00,D0,A9,88,8D,01,D0, 1020 <079>
1421 DATA A9,04,85,58,A9,D8,85,5A,A2, 1164 <113>
1422 DATA 00,86,57,86,59,A9,19,85,FB, 1022 <086>
1423 DATA A0,00,BD,C0,02,8D,BA,1F,BD, 1090 <149>
1424 DATA C1,02,8D,BB,1F,BD,C2,02,8D, 1080 <147>
1425 DATA BC,1F,A9,06,91,59,C0,18,B0, 1020 <106>
1426 DATA 1C,A5,FB,C9,05,90,16,AD,93, 1136 <133>
1427 DATA 1F,49,80,91,57,2E,BC,1F,2E, 775  <086>
1428 DATA BB,1F,2E,BA,1F,90,04,A9,04, 802  <089>
1429 DATA 91,59,C8,C0,28,D0,D7,E8,E8, 1553 <144>
1430 DATA E8,A5,57,18,69,28,85,57,85, 1006 <088>
1431 DATA 59,90,04,E6,58,E6,5A,C6,FB, 1324 <138>
1432 DATA D0,AD,A9,06,8D,27,D0,A5,5B, 1200 <142>
1433 DATA 8D,BA,1F,A5,5C,8D,BB,1F,A2, 1136 <198>
1434 DATA 00,20,FA,19,60,20,57,0F,AD, 710  <040>
1435 DATA 18,D0,29,F7,8D,18,D0,AD,11, 1083 <128>
1436 DATA D0,29,DF,8D,11,D0,20,CC,FF, 1329 <174>
1437 DATA A9,93,20,D2,FF,A9,0F,8D,21, 1171 <148>
1438 DATA D0,A9,06,8D,86,02,A9,00,85, 962  <058>
1439 DATA C6,8D,15,D0,60,20,E1,1C,A2, 1111 <107>
1440 DATA 00,A0,00,B9,F0,1C,85,57,F0, 1073 <100>
1441 DATA 27,C9,FF,D0,04,E8,C8,D0,F1, 1588 <183>
1442 DATA B9,F1,1C,9D,40,03,E8,B9,F2, 1337 <158>
1443 DATA 1C,9D,40,03,E8,B9,F3,1C,9D, 1097 <162>
1444 DATA 40,03,E8,C6,57,D0,E7,C8,C8, 1423 <145>
1445 DATA C8,C8,D0,D2,AD,18,D0,09,08, 1240 <147>
1446 DATA 8D,18,D0,AD,11,D0,09,20,8D, 953  <086>
1447 DATA 11,D0,20,48,0F,AD,20,D0,8D, 898  <085>
1448 DATA 27,D0,AD,84,1F,4C,07,0E,A9, 849  <115>
1449 DATA 00,A0,3F,99,C0,02,88,10,FA, 972  <072>
1450 DATA 60,AA,F0,0A,A0,00,EA,EA,88, 1280 <165>
1451 DATA D0,FB,CA,D0,F8,60,A5,CB,C9, 1782 <224>
1452 DATA 40,D0,FA,60,20,A2,12,29,10, 887  <048>
1453 DATA D0,F9,60,08,00,20,00,02,00, 595  <009>
1454 DATA 00,00,01,FF,07,F8,02,00,00, 513  <015>
1455 DATA 00,08,00,20,00,FF,06,00,00, 301  <243>
1456 DATA 00,01,03,FF,00,08,02,01,00, 270  <248>
1457 DATA 01,03,FF,00,05,00,00,00,FF, 519  <037>
1458 DATA 01,FF,FF,FF,13,C0,00,03,01, 981  <104>
1459 DATA FF,FF,FF,00,20,E1,1C,A9,20, 1251 <188>
1460 DATA 8D,93,1F,A9,00,85,5B,85,5C, 937  <109>
1461 DATA 8D,83,1F,20,47,1C,20,B6,1B, 675  <087>
1462 DATA 20,55,1D,20,2E,1E,20,7D,1E, 441  <070>
1463 DATA A9,30,20,D3,1C,A5,CB,C9,3C, 1117 <172>
1464 DATA D0,EC,4C,74,1C,A5,CB,CD,80, 1365 <215>
1465 DATA 1F,D0,01,60,8D,80,1F,C9,24, 873  <094>
1466 DATA D0,25,A0,3C,B9,C0,02,20,0A, 886  <089>
1467 DATA 1E,99,BC,1F,B9,C2,02,20,0A, 825  <119>
1468 DATA 1E,99,BA,1F,B9,C1,02,20,0A, 822  <114>
1469 DATA 1E,99,BB,1F,88,88,88,10,E0, 1049 <168>
1470 DATA 4C,19,1E,C9,16,D0,14,A0,00, 742  <083>
1471 DATA A2,3E,B9,C0,02,20,0A,1E,9D, 832  <113>
1472 DATA BA,1F,C8,CA,10,F3,4C,19,1E, 1009 <200>
1473 DATA C9,11,D0,44,A9,02,85,FB,A4, 1213 <156>
1474 DATA FB,84,FD,BE,28,1E,B9,2B,1E, 1154 <235>
1475 DATA 85,FA,A9,03,85,FC,A9,08,85, 1250 <181>
1476 DATA FE,A4,FA,8A,48,A9,00,3E,C0, 1301 <206>
1477 DATA 02,6A,E8,E8,E8,88,D0,F6,A4, 1558 <209>
1478 DATA FD,99,BA,1F,C8,C8,C8,84,FD, 1608 <006>
1479 DATA 68,AA,C6,FE,D0,E0,E8,C6,FC, 1840 <004>
1480 DATA D0,D7,C6,FB,10,C3,4C,19,1E, 1214 <194>
1481 DATA C9,1A,D0,08,AD,93,1F,49,6F, 981  <164>
1482 DATA 8D,93,1F,4C,B6,1B,C9,33,D0, 1064 <192>
1483 DATA 0D,AD,8D,02,29,01,F0,06,20, 649  <099>
1484 DATA C8,1C,4C,B6,1B,60,85,FE,84, 1128 <199>
1485 DATA FD,A0,08,26,FE,6A,88,D0,FA, 1413 <226>
1486 DATA A4,FD,60,A0,3E,B9,BA,1F,99, 1290 <227>
1487 DATA C0,02,88,10,F7,20,B6,1B,60, 930  <085>
1488 DATA 27,0F,00,08,08,05,20,A2,12, 287  <049>
1489 DATA A0,00,29,0F,F0,35,4A,A6,5B, 840  <116>
```

Listing 2. HI-EXE (Fortsetzung)

```
1490 DATA F0,04,90,02,C6,5B,4A,90,06, 903  <091>
1491 DATA E0,14,B0,02,E6,5B,4A,A6,5C, 1075 <184>
1492 DATA F0,04,90,02,C6,5C,4A,90,06, 904  <095>
1493 DATA E0,17,B0,02,E6,5C,AD,93,1F, 1098 <199>
1494 DATA 49,80,91,57,20,37,1C,AD,93, 868  <108>
1495 DATA 1F,A0,00,91,57,AD,83,1F,49, 831  <121>
1496 DATA 01,8D,83,1F,F0,06,B1,57,49, 887  <122>
1497 DATA 80,91,57,60,AD,82,1F,29,10, 847  <102>
1498 DATA F0,3C,20,F7,14,A5,5C,4A,4A, 1004 <187>
1499 DATA 4A,A0,03,18,65,5B,88,D0,FB, 1048 <188>
1500 DATA A8,A5,5E,AE,8D,02,D0,0B,19, 988  <182>
1501 DATA C0,02,99,C0,02,A9,04,4C,B1, 967  <123>
1502 DATA 1E,49,FF,39,C0,02,99,C0,02, 956  <141>
1503 DATA A9,06,A8,A5,5B,AA,49,DC,85, 1192 <223>
1504 DATA 5B,98,A0,00,91,57,86,5B,60, 950  <088>
1505 DATA A0,03,D9,22,1F,F0,03,88,D0, 1032 <164>
1506 DATA F8,C8,98,0A,0A,AA,A0,03,CA, 1155 <232>
1507 DATA BD,26,1F,99,B6,1F,88,10,F6, 1022 <207>
1508 DATA AD,80,1F,4A,4A,4A,8D,B4,1F, 906  <195>
1509 DATA A9,04,AE,B3,1F,AC,B4,1F,20, 972  <190>
1510 DATA BA,FF,A0,02,B1,2D,48,C8,B1, 1274 <237>
1511 DATA 2D,AA,C8,B1,2D,A8,68,20,BD, 1130 <235>
1512 DATA FF,AE,B6,1F,86,57,AC,B7,1F, 1249 <027>
1513 DATA B4,5B,AD,B4,1F,D0,06,A9,00, 987  <164>
1514 DATA 20,D5,FF,60,A9,57,AE,B8,1F, 1241 <240>
1515 DATA AC,B9,1F,20,D8,FF,60,47,46, 1128 <230>
1516 DATA 53,5A,00,20,00,40,00,20,00, 301  <026>
1517 DATA 44,C0,02,FF,02,00,20,00,28, 591  <087>
1518 DATA A2,03,20,C6,FF,20,CF,FF,20, 1176 <231>
1519 DATA D2,FF,D0,F8,A9,0D,20,D2,FF, 1600 <016>
1520 DATA 20,CC,FF,60,FF,9C,          998  <012>
```

Listing 2. HI-EXE (Schluß)

Listing 1. HI-PRINT ist die ausgezeichnete Drucker-Routine des HI-EDDI. Alle Hardcopies auf diesen Seiten wurden mit HI-PRINT erstellt. Auch dieses Unterprogramm speichert sich nach dem Starten als Maschinenprogramm auf Diskette. Beachten Sie den Beitrag »Checksummer« in dieser Ausgabe. Er gibt Ihnen wichtige Hinweise zum Abtippen.

```
10 REM********************************   <217>
20 REM*                              *   <247>
30 REM*           HI-EDDI            *   <213>
40 REM*       VON HANS HABERL        *   <214>
50 REM*                              *   <021>
60 REM*   DATA-LADER ZUR ERZEUGUNG   *   <139>
70 REM*DER DRUCKERROUTINE "HI-PRINT",*   <245>
80 REM*  WIRD AUF DISKETTE ABGELEGT  *   <221>
90 REM********************************   <041>
100 :                                    <158>
110 REM  CHECKSUMMEN, NICHT VERTIPPEN !! <209>
120 DATA-205,164,-457,322,689,-834,-49,290,-87,
   17                                    <246>
130 :                                    <188>
140 REM M-CODE PROGRAMM                  <021>
150 DATA 0,13,133,91,173,105,14,240,34,120,169,
   255,141,3,221,173,2,221,9,4           <129>
160 DATA 141,2,221,173,0,221,9,4,141,0,221,169,
   16,141,13,221,173,13,221,88,-1        <016>
170 DATA 76,68,13,32,204,255,169,4,174,106,14,
   172,107,14,32,186,255,169,0           <142>
180 DATA 32,189,255,32,192,255,162,4,32,201,255,
   169,4,133,94,160,3,32,34,14,-1        <121>
190 DATA 169,128,133,95,169,2,133,96,165,91,41,
   192,208,4,70,95,70,96,165,91          <035>
200 DATA 41,7,162,0,32,48,14,165,91,48,3,74,74,
   74,41,7,162,2,32,48,14,169,-1         <047>
210 DATA 25,133,92,165,95,48,10,160,20,169,32,
   32,71,14,136,208,250,160,8,32         <008>
220 DATA 34,14,165,95,32,71,14,165,96,32,71,14,
   169,40,133,93,120,169,52,133,-1       <170>
230 DATA 1,160,7,177,87,153,125,14,136,16,248,
   169,55,133,1,88,165,87,24,105         <056>
240 DATA 8,133,87,144,2,230,88,160,8,162,0,62,
   125,14,8,42,232,40,36,91,48,-1        <080>
250 DATA 6,224,8,208,241,240,8,42,228,94,208,
   234,32,71,14,32,71,14,136,208         <000>
260 DATA 223,198,93,208,186,165,95,16,38,162,1,
   180,87,181,89,149,87,148,89,-1        <220>
270 DATA 202,16,245,165,94,73,12,133,94,201,4,
   240,15,165,91,48,3,76,151,13          <022>
280 DATA 160,0,32,34,14,76,122,13,198,92,208,
   244,160,0,32,34,14,160,13,32,-1       <097>
290 DATA 34,14,32,204,255,169,4,32,195,255,96,
   185,108,14,201,255,240,6,32           <000>
300 DATA 71,14,200,208,243,96,160,6,217,150,31,
   240,3,136,208,248,169,0,149,-1        <189>
310 DATA 87,24,105,32,136,16,251,149,88,96,174,
   105,14,208,3,76,210,255,72            <033>
320 DATA 141,1,221,173,0,221,41,251,141,0,221,9,
   4,141,0,221,173,13,221,41,-1          <103>
330 DATA 16,240,249,104,96,-1            <168>
340 :                                    <143>
350 REM DIE FOLGENDEN DATAS MUESSEN AN   <110>
360 REM DEN DRUCKER ANGEPASST WERDEN!!   <140>
370 REM (DIE ANGEGEBENEN WERTE SIND      <192>
380 REM FUER EINEN EPSON RX-80 MIT       <091>
390 REM DATA BECKER INTERFACE)           <150>
400 :                                    <203>
410 REM 0=SER. BUS, 1=USERPORT           <183>
420 DATA 0                               <088>
430 REM GERAETEADRESSE                   <066>
440 DATA 4                               <112>
450 REM SEKUNDAERADRESSE (DIREKTMODUS)   <151>
460 DATA 1                               <129>
470 REM DIE LAENGE DER FOLGENDEN DATA-   <152>
480 REM ZEILEN DARF NICHT VERAENDERT     <186>
490 REM WERDEN, GGF. MIT 255 AUFFUELLEN  <218>
500 REM (MIND. EIN 255 MUSS IN JEDER     <195>
510 REM ZEILE BLEIBEN, IST ENDEKENNZ.)   <013>
520 REM CARRIAGE RETURN LINE FEED        <243>
530 DATA 13,10,255                       <080>
540 REM ZEILENABSTAND FUER GRAFIK        <087>
550 DATA 27,51,23,255,255                <199>
560 REM CRT-GRAFIK (640 PUNKTE/ZEILE)    <245>
570 REM (OHNE BYTE-ANZAHLEN!)            <025>
580 DATA 27,42,4,255,255                 <180>
590 REM NORMALER ZEILENABSTAND           <003>
600 DATA 27,50,255,255                   <103>
610 DATA-2:REM DATA-ENDE                 <114>
620 :                                    <168>
630 REM CHECKSUMMENPRUEFUNG              <166>
640 V=1:FOR I=0 TO 9:READ S(I):NEXT      <232>
650 FOR B=0 TO 9                         <014>
660 READ A:IF A>=0 THEN S=S+A*V:V=-V:GOTO 660
                      <133>
670 IF S<>S(B)THEN PRINT"DATA-FEHLER IN ZEILE"1
   50+20*B"ODER"160+20*B:END             <162>
680 S=0:NEXT:PRINT"DATAS OK"             <185>
690 :                                    <238>
700 REM FILE-ABLAGE                      <054>
710 RESTORE:FOR I=0 TO 9:READ A:NEXT     <213>
720 OPEN 2,8,2,"HI-PRINT,P,W"            <031>
730 READ A:IF A>=0 THEN PRINT#2,CHR$(A); <110>
740 IF A>-2 THEN 730                     <129>
750 CLOSE 2:END                          <124>
```

Listing 3. HI-EDDI ist sozusagen das Steuerprogramm. Es ruft die anderen benötigten Programme auf. Tippen Sie die REM-Zeilen nicht mit ab.

```
1 REM ZEILEN WEGLASSEN !!                          <066>
2 :                                                <060>
3 REM ********************************             <252>
4 REM *                                  *         <231>
5 REM *             HI-EDDI              *         <188>
6 REM *                                  *         <233>
7 REM *   VON:   HANS HABERL             *         <239>
8 REM *          BAHNHOFSTR. 3           *         <059>
9 REM *          8018  GRAFING           *         <187>
10 REM*          TEL. 08092/7152         *         <001>
11 REM*                                 *          <238>
12 REM*********************************            <005>
13 :                                               <071>
50 IF A<2 THEN A=A+2:LOAD"HI-EXE",8,1              <012>
60 IF A=3 THEN 150                                 <042>
70 IF A=4 THEN 650                                 <058>
100 INPUT"BETRIEBSART";C:POKE 780,C:POKE 56,13
   :CLR:W$="123456":SYS 3328                       <235>
110 IF PEEK(8073)AND 64 THEN A=3:LOAD"MENUE",8,1
                         <016>
150 SYS 3337:A=PEEK(8064)                          <022>
160 ON A+1 GOSUB 200,250,300,290,400,600,290,
   290,500,510,300:GOTO 150                        <096>
200 OPEN 3,8,0,"$0":GET#3,C$,C$                    <117>
210 GET#3,C$,C$,L$,H$:IF C$=""THEN CLOSE 3
   :WAIT 198,1:RETURN                              <082>
220 PRINT 256*ASC(H$+CHR$(0))+ASC(L$+CHR$(0));
   :SYS 3334:GOTO 210                              <140>
250 PRINT"[CLR,DOWN]SEQUENZ:"
   :PRINT"[DOWN,2SPACE]"W$:INPUT"[HOME,3DOWN]";
   W$:IF LEN(W$)<2 THEN 250                        <249>
290 RETURN                                         <177>
300 POKE 8115,8:PRINT"[CLR,DOWN]G=GRAFIKBILD"
   :IF PEEK(8073)>127 THEN PRINT"F=FARBBILD"
                         <055>
310 PRINT"S=SPRITE":PRINT"Z=ZEICHENSATZ"
   :PRINT"[DOWN,SPACE]-->";                        <108>
320 GET C$:IF C$<>"G"AND C$<>"F"AND C$<>"S"AND
   C$<>"Z"THEN 320                                 <205>
330 PRINT C$:POKE 780,ASC(C$):H$=W$
   :INPUT"[DOWN]FILENAME";W$:SYS 3331:W$=H$
   :H$=""                                          <084>
340 OPEN 1,8,15,H$:INPUT#1,A,H$:PRINT A;H$
   :IF A>0 THEN WAIT 198,1                         <129>
350 CLOSE 1:RETURN                                 <248>
400 POKE 631,34:POKE 198,1:H$=""
   :INPUT"[CLR,DOWN]";H$:GOTO 340                  <105>
500 A=8075:GOTO 520                                <022>
510 A=8079                                         <202>
520 D=PEEK(8074):C=PEEK(A+D)
   :PRINT"[CLR,DOWN]F"2*D+1:PRINT"SCHRITTWEITE
   :"C,                                            <180>
530 INPUT C:IF C<1 OR C>160 THEN 530               <066>
540 POKE A+D,C:RETURN                              <027>
600 INPUT"[CLR,DOWN]NUMMER 1.BILD";D:C=(D AND 7)
                         <062>
610 INPUT"[DOWN]NUMMER 2.BILD (0=NUR 1 BILD)";D
   :C=C+8*(D AND 7):IF D>0 THEN C=C+64             <099>
620 IF D=0 THEN PRINT"[DOWN]GROSS";:GOSUB 800
   :IF C$="J"THEN C=C+128                          <108>
630 PRINT"[DOWN]DRUCKEN";:GOSUB 800
   :IF C$="N"THEN 660                              <191>
640 IF A=5 THEN A=4:LOAD"HI-PRINT",8,1             <032>
650 POKE 780,C:SYS 3328:PRINT"[DOWN]NOCHMAL";
   :GOSUB 800:IF C$="J"THEN 600                    <070>
660 IF A=5 THEN RETURN                             <126>
670 A=1:GOTO 50                                    <236>
800 PRINT" ? (J/N)";                               <114>
810 GET C$:IF C$<>"J"AND C$<>"N"THEN 810           <159>
820 PRINT C$:RETURN                                <255>
```

Listing 4. Ein Beispiel für den Aufbau einer Steuertafel für ein selbsterstelltes Menü. Einzelheiten finden Sie in der Beschreibung.

```
10 REM*********************************            <003>
20 REM*                                *           <247>
30 REM*           HI-EDDI              *           <213>
40 REM*       VON HANS HABERL          *           <214>
50 REM*                                *           <021>
60 REM*  DATA'S FUER DIE STEUERZEILE   *           <008>
70 REM*    DER BEISPIEL-MENUETAFEL     *           <100>
80 REM*********************************            <073>
90 REM JE ZWEI AUFEINANDERFOLGENDE DATA-ZEILEN
                         <043>
100 REM ERGEBEN EINE ZEILE IN DER MENUETAFEL (=
   20 FELDER)                                      <127>
110 DATA 64,0,64,0,64,0,64,0,64,0,64,0,64,0,64,
   0,64,0,64,0                                     <057>
120 DATA 64,0,64,0,64,0,64,0,64,0,64,0,64,0,64,
   0,64,0,64,0                                     <067>
130 DATA 0,18,0,18,0,18,0,20,0,20,0,20,0,22,0,
   22,0,22                                         <046>
140 DATA 0,13,0,13,0,13,0,10,0,10,0,10,0,56,56,
   23,56,38,56,30,1,51                             <192>
150 DATA 0,18,0,18,0,18,0,20,0,20,0,20,0,22,0,
   22,0,22                                         <066>
160 DATA 0,13,0,13,0,13,0,10,0,10,0,10,0,59,59,
   23,59,38,59,30,1,51                             <224>
170 DATA 0,42,0,42,0,42,0,41,0,41,0,41,0,9,0,9,
   0,9                                             <213>
180 DATA 0,26,0,26,0,26,0,14,0,14,0,14,0,8,8,23,
   8,38,8,30,1,51                                  <052>
190 DATA 0,42,0,42,0,42,0,41,0,41,0,41,0,9,0,9,
   0,9                                             <233>
200 DATA 0,26,0,26,0,26,0,14,0,14,0,14,0,11,11,
   23,11,38,11,30,0,33                             <239>
210 DATA 0,17,0,17,0,17,0,36,0,36,0,36,1,9,1,9,
   1,9                                             <018>
220 DATA 1,60,1,60,1,60,1,60,1,60,1,60,0,16,16,
   23,16,38,16,30,0,33                             <026>
230 DATA 0,17,0,17,0,17,0,36,0,36,0,36,1,9,1,9,
   1,9                                             <038>
240 DATA 1,60,1,60,1,60,1,60,1,60,1,60,0,19,19,
   23,19,38,19,30,0,33                             <058>
250 DATA 128,21,132,21,136,21,140,21,144,21,148,
   21,152,21,156,21,160,21                         <069>
260 DATA 164,21,168,21,172,21,176,21,180,21,184,
   21,188,21,0,4,0,5,0,6,0,3                       <180>
270 DATA 128,28,132,28,136,28,140,28,144,28,148,
   28,152,28,156,28,160,28                         <153>
280 DATA 164,28,168,28,172,28,176,28,180,28,184,
   28,188,28,1,4,1,5,1,6,1,3                       <253>
290 DATA 129,21,133,21,137,21,141,21,145,21,149,
   21,153,21,157,21,161,21                         <119>
300 DATA 165,21,169,21,173,21,177,21,181,21,185,
   21,189,21,2,4,2,5,2,6,2,3                       <235>
310 DATA 129,28,133,28,137,28,141,28,145,28,149,
   28,153,28,157,28,161,28                         <202>
320 DATA 165,28,169,28,173,28,177,28,181,28,185,
   28,189,28,0,29,0,29,0,31,0,31                   <244>
330 DATA 2,18,2,18,2,18,2,18,2,42,2,42,2,42,2,
   42,2,13,2,13,2,13,2,13                          <230>
340 DATA 2,20,2,20,2,20,2,20,2,41,2,41,2,41,2,41
                         <024>
350 REM  KENNUNG "CBM80" ,ALS MENUEKENNUNG MISS
   BRAUCHT                                         <099>
360 DATA 195,194,205,56,48                         <071>
370 :                                              <173>
380 POKE 56,32:CLR:REM IM SPEICHER BEFINDLICHE
   MENUETAFEL SCHUETZEN                            <068>
390 FOR I=15872 TO 15872+484:READ A:POKE I,A
   :S=S+A:NEXT I                                   <010>
400 IF S<>18329 THEN PRINT"DATA-FEHLER!"           <124>
```

Wie schicke ich meine Programme ein?

Die 64'er-Redaktion freut sich über jeden Beitrag unserer Leser. Die Erfahrung hat aber gezeigt, daß viele Einsender nicht genau wissen, in welcher Form sie ihre Manuskripte einsenden sollen. Die unten aufgeführten Punkte stellen keine »Richtlinien« dar. Dennoch sollte sich jeder, der ein Programm oder einen Artikel einsenden will, an ein gewisses Schema halten. Dies erleichtert zum einen die Arbeit der Redaktion, zum anderen konnt es auch Ihnen selbst zugute, da wir vollständige Listings oder Artikel schneller veröffentlichen können. Folgende Kriterien sind also generell zu beachten.

1. Auf der ersten Seite des Anschreibens sollten der Name, die vollständige Anschrift mit Telefonnummer sowie das Einsendedatum stehen.

2. In der »Betreffzeile« tragen Sie die genaue Spezifikation des verwendeten Computers und falls erforderlich, die Basic-, ROM- oder DOS-Versionen sowie die Speicherkonfigurationen ein. Der Titel des Artikels sollte ebenfalls daraus ersichtlich sein (auch für eventuelle Nachträge).

3. Im darauffolgenden Text können Sie Wesentliches zu Ihrer Person, zur Entstehungsgeschichte des Programms/Artikels, der Absicht, der Vorteile gegenüber anderen Programmen oder Methoden, der Eigenschaften und so weiter erläutern.

4. Auf der nächsten Seite beginnt die eigentliche Programmbeschreibung. Diese sollte nach Möglichkeit mit der Schreibmaschine geschrieben werden oder als Computerausdruck vorliegen. Den Text bitte mit mindestens eineinhalb oder doppeltem Zeilenabstand verfassen. Am linken und rechten Rand mindestens drei Zentimeter Freiraum für Korrekturen und Bemerkungen lassen.

5. Diese und alle nachfolgenden Seiten sollten durchnumeriert sein und in der Kopfzeile jeweils den Titel des Programms und den Namen des Autors enthalten.

6. Der Überschrift des Artikels schließen sich zwei oder drei einleitende Sätze an, welche die wesentlichen Punkte des Textes zusammenfassen.

Der Text selbst sollte in etwa folgenden Aufbau aufweisen:
— Angaben auf welchem Computer das Programm lauffähig ist sowie welche Erweiterungen und Peripherie notwendig sind
— ausführliche Beschreibung der Programmfunktion (mit Verweisen auf Ein-/Ausgabebeispielen wie Grafiken, Bildschirmfotografien, Hardcopys oder Diagrammen)
— detaillierte Programmbeschreibung (mit Verweisen auf Programmablaufplan, Variablendefinition, Startadressen der einzelnen Unterprogramme, Beschreibung wichtiger Programmzeilen etc.)
— eventuelle Umsetzung auf andere Basic-Dialekte oder Computer

7. Die genauen Lade- und Abspeicherschritte des Programms und der im Programm vorkommenden Routinen sollten dokumentiert sein.

8. Listings aus reprotechnischen Gründen nur als Original (keine Kopien) auf weißem, unliniertem Papier mit neuwertigem Farbband gedruckt einsenden. In den Listings dürfen grundsätzlich keine handschriftlichen Eintragungen stehen.

9. In den Kopfzeilen des Programms bitte den Titel desselben, die Computerkonfiguration, den eigenen Namen und die Adresse mit Telefonnummer eintragen (es soll vorkommen, daß sich Listings und Manuskripte verselbständigen, und mit beiden allein läßt sich wenig anfangen).

REM-Zeilen im Programm dienen der Übersichtlichkeit und sollten, falls nicht speicherkritische Aspekte dagegensprechen, immer zur Strukturierung eingesetzt werden (siehe u. a. »Sauberes Programmieren«).

10. Um das Eintippen für andere zu erleichtern, sollten Sie CHR$(X)-Werte und TAB(X) oder SPC(X) anstatt Cursor-Manipulationen für die Ausgabeformatierung verwenden. So ist die Befehlssequenz FOR I=1 TO 6:PRINT:NEXT zur Erzeugung von sechs Carriage Returns leichter einzutippen und auf andere Basic-Computer wesentlich einfacher zu übertragen. Und ist es nicht auch übersichtlicher statt einem Dutzend Cursor-Rechts-Symbolen einfach SPC(12) zu benutzen? Überprüfen Sie Ihr Programm einmal hinsichtlich dieser »Kleinigkeiten«.

11. Da wir (in Ihrem eigenen Interesse) nur getestete Programme veröffentlichen wollen, legen Sie bitte unbedingt eine Diskette oder Kassette, auf der das betreffende Programm mit mindestens einer Sicherheitskopie abgespeichert ist, bei. Auf der Diskette/Kassette und deren Umhüllung unbedingt den Namen mit vollständiger Adresse und Computerbezeichnung vermerken.

12. Wollen Sie mehrere Programme/Artikel gleichzeitig einsenden, so trennen Sie die Programme/Artikel nach dem oben aufgezeigten Schema. Die Einsendung mehrerer Disketten/Kassetten ist hingegen nicht notwendig.

13. Artikel können beliebig lang sein — von einzeiligen Routinen bis zu Serien über mehrere Ausgaben. Ein durchschnittlicher Artikel hat rund vier bis acht Schreibmaschinenseiten.

14. Hardcopys, Flußdiagramme, Zeichnungen und Bildschirmfotos dienen der Anschaulichkeit. Sie sollten nach Möglichkeit nicht fehlen. Zu jedem der vorgenannten »Zugaben« gehört aber eine Bildunterschrift und ein Verweis im Text.

15. Programme/Artikel die unserem Verlag zur Veröffentlichung angeboten werden, sollten aus urheberrechtlichen Gründen nicht gleichzeitig einem anderem Verlag vorliegen.

16. Das 64'er Magazin zahlt für Listings eine Pauschale zwischen 100 und 300 Mark. Für reine Artikel beträgt das Honorar zwischen 0,80 und 1,00 Mark pro Druckzeile. Für Disketten/Kassetten werden 30 Mark extra berechnet.

17. Sollten sich nach Erhalt eines positiven Anwortschreibens noch irgendwelche Änderungen oder Verbesserungen des Programms ergeben haben, teilen Sie uns das bitte umgehend mit. In diesem Falle benötigen wir ein vollständig neues Listing mit entsprechendem Datenträger.

(aa)

Ohne gutes Werkzeug geht es nicht:

SMON, Teil 3

Der Maschinensprache-Monitor geht langsam seiner Vollendung entgegen. In diesem Teil kommen drei interessante Befehle hinzu, die vor allem bei der Fehlersuche sehr hilfreich sind.

Sicherlich haben Sie sich beim letzten Mal gewundert, wie es möglich ist, daß so viele neue Befehle in so wenig Programm stecken können. »Schuld« daran ist das SMON-Konzept. Wir haben im ersten Teil bereits alle Ein- und Ausgaberoutinen untergebracht. Alle Erweiterungen können nun darauf aufbauen und werden dementsprechend kürzer. Wir haben Ihnen sogar noch einen Befehl verschwiegen, der beim letzten Mal schon vorhanden war. Vielleicht haben Sie versehentlich einmal »B« eingegeben, und SMON hat mit einem Fragezeichen reagiert und damit gezeigt, daß er mit »B« etwas anzufangen weiß.

Im heutigen Teil 3 unserer SMON-Serie wollen wir Ihnen drei weitere Befehle vorstellen: BASIC-DATA, KONTROLLE und FIND. Es sind diesmal nur drei neue Befehle, nicht weil wir Sie für den nächsten Artikel über »SMON« auf die Folter spannen wollen, sondern weil wir der Meinung sind, daß ein so umfassender Befehl wie »FIND« schon eine Menge an Beispielen braucht, um verstanden zu werden.

BASIC-DATA

B (ANFADR ENDADR) wandelt das Maschinenprogramm von ANFADR bis ENDADR-1 in Basic-DATA-Zeilen um.
B 4000 4020
Unser Testprogramm (Sie erinnern sich doch noch an unser kleines Programm aus 11/84?) wird in DATA-Werte umgerechnet und dann mit Zeilennummer 32000 beginnend im Basic-Speicher abgelegt. Ein im Speicher befindliches Basic-Programm (zum Beispiel ein Basic-Lader) mit kleineren Zeilennummern kann dann diese DATA-Zeilen benutzen.

Wenn Sie das Testprogramm wie oben beschrieben umgewandelt haben, verlassen Sie nun mit »X« den SMON und überzeugen sich mit »LIST« von der Ausführung. Dann können Sie folgendes eingeben:
10 FOR I=16384 TO 16415 : READ D :POKE I,D : NEXT

In Verbindung mit den oben erzeugten DATA-Zeilen (und RUN!) hätten Sie wieder das ursprüngliche Maschinenprogramm im Speicher. Falls Sie dieses Beispiel durchführen wollen, denken Sie bitte daran, daß Sie nach Erstellung der DATAs, das Originalprogramm zum Beispiel mit OCCUPY (O 4000 4020 AA) überschreiben, damit Sie die richtige Ausführung überprüfen können. Der BRK-Befehl am Ende des Testprogramms bewirkt einen Sprung zum SMON zurück. Wollen Sie ein Maschinenprogramm von Basic aus starten und auch wieder dorthin zurückgelangen, muß der letzte Befehl ein RTS sein. Probieren Sie es aus, indem Sie das Basic-Programm um 20 SYS 16384 erweitern.

KONTROLLE

K (ANFADR ENDADR) listet die ASCII-Zeichen im gewünschten Bereich. Es werden jeweils 32 Zeichen pro Zeile ausgegeben, so daß man sich einen schnellen Überblick über Texte oder Tabellen verschaffen kann.

Beispiel:
K 4000 listet die ersten 32 Zeichen unseres Programms. Die weitere Ausgabe ist genau wie beim Disassemblieren durch Druck auf SPACE oder RETURN möglich. Auch hier können Sie wie bei den anderen Bildschirm-Ausgabebefehlen Änderungen durch einfaches Überschreiben vornehmen (natürlich nicht im ROM und nur mit ASCII-Zeichen!).

Als Beispiel wollen wir einmal im Basic »herumpfuschen«: Das geht natürlich nicht so ohne weiteres, weil das Basic im ROM steht und damit nicht verändert werden kann. Tippen Sie bitte folgendes ein:
W A000 C000 A000
Auf den ersten Blick eine unsinnige Anweisung; der Speicher soll von A000 bis C000 nach A000 verschoben werden. Dieser Befehl entspricht exakt der Basic-Schleife
FOR I = 40960 TO 49152 : POKE I, PEEK (I) : NEXT

Nun ist es aber so, daß beim PEEK das ROM gelesen, beim POKE aber ins darunterliegende RAM geschrieben wird. Wir erreichen also, daß das Basic ins RAM kopiert wird. Jetzt müssen wir dafür sorgen, daß das Betriebssystem sein Basic aus dem RAM und nicht aus dem ROM holt. Zuständig dafür ist die Speicherstelle 0001. Geben Sie bitte »M 0001« ein, und überschreiben Sie die »37« mit »36«.

Es passiert gar nichts. Jetzt tritt unser K-Kommando in Aktion. Geben Sie ein: K A100 A360
Was Sie sehen, sind die Basic-Befehlswörter und -Meldungen. Schalten Sie mit SHIFT/CBM auf Kleinschrift, dann erkennen Sie, daß der jeweils letzte Buchstabe eines Befehlswortes groß geschrieben ist (Endekennung). Jetzt ändern Sie durch Überschreiben das »LIST« (A100) in »LUST« und »ERROR« (A360) in »FAELER«. (Bei »FAELER« müssen Sie ein Zeichen vor »ERROR« beginnen, sonst paßt es nicht.)

Verlassen Sie jetzt SMON mit »X« und geben Sie danach ein:
POKE 1,54
SMON schaltet nämlich beim »X«-Befehl immer auf das Basic-ROM zurück, daher müssen wir wieder auf unser geändertes Basic umschalten. Schreiben Sie nun einen Basic-Dreizeiler und versuchen Sie, diesen zu LISTen. Ergebnis? Versuchen Sie es jetzt einmal mit »LUST«. Ihrer weiteren Phantasie sind keine Grenzen mehr gesetzt...
Wie oben angesprochen stellt SMON eine Reihe verschiedener Suchroutinen zur Verfügung, die im Folgenden an vielen Beispielen beschrieben werden. Alle diese Befehle bestehen aus zwei Zeichen und beginnen mit dem Buchstaben »F«.

FIND

F (HEX-WERT(e), ANFADR ENDADR) sucht nach einzelnen HEX-Werten innerhalb eines bestimmten Bereichs. Das zweite Zeichen (hinter F) ist hier ein Leerzeichen und darf nicht weggelassen werden! Die Bereichsangabe kann wie bei allen folgenden Befehlen entfallen, dann wird der gesamte Speicher durchsucht.
Beispiel: Wir suchen alle Befehle LDY #01, also die Werte A0 01 im Bereich von $2000 bis $6000.
F A0 01, 2000 6000 (Die Leerzeichen zwischen den Hexbytes dürfen nicht weggelassen werden !). Es erscheinen alle Speicherstellen, die die gesuchten Bytes enthalten, also zum Beispiel 4000.
FA (Adresse, ANFADR ENDADR) sucht alle Befehle, die eine bestimmte Adresse als Operanden haben (absolut). Die Adresse braucht nicht vollständig angegeben zu werden, es kann das Jokerzeichen »*« benutzt werden.
1. Beispiel: Wir suchen alle JSR FFD2-Befehle im Bereich $2000 bis $6000.
FAFFD2,2000 6000
Es erscheinen alle Befehle disassembliert, die FFD2 im Operanden enthalten (also auch LDA FFD2 oder STA FFD2,Y...).
2. Beispiel: Wir suchen alle Befehle, die auf den Grafikbereich ($D000 bis $DFFF) zugreifen.
FAD***,2000 6000
Der Joker kann aber auch zum Beispiel zur Suche im Bereich $D000 bis $D0FF dienen: FAD0**,2000 6000
FR (ADR, ANFADR ENDADR) sucht nach relativen Sprungzielen. Anders als bei absoluten Sprüngen (JMP, JSR) benutzen die Branch-Befehle eine relative Adressierung, also zum Beispiel »Verzweige 10 vor« oder »37 zurück«. Solche Sprünge lassen sich mit dem FA-Kommando nicht finden. Hier wird »FR« eingesetzt.
Beispiel: Gesucht werden alle Branch-Befehle, die die Adresse $4002 anspringen.
FR4002,2000 6000
Natürlich können solche Befehle nur höchstens 128 Byte vom Sprungziel entfernt sein. Die Bereichsangabe ist hier also viel zu groß gewählt (SMON stört dies allerdings nicht). Der Einsatz des Jokers ist hier ebenfalls wie oben beschrieben möglich.
FT (ANFADR ENDADR) sucht Tabellen im angegebenen Bereich. SMON behandelt dabei alles, was sich nicht disassemblieren läßt, als Tabelle.
Beispiel: Wir suchen Tabellen oder Text im Bereich $2000 bis $6000.
FT 2000 6000
FZ (Adr, ANFADR ENDADR) sucht alle Befehle, die Zeropage-Adressen haben.
1. Beispiel: FZC5,2000 6000 findet alle Befehle, die C5 adressieren, also zum Beispiel BIT $C5, LDA (C5), Y etc.
2. Beispiel: FZF*,2000 6000 findet alle Befehle, die den Bereich zwischen $F0 und $FF adressieren.
3. Beispiel: FZ**,2000 6000 findet sämtliche Befehle mit Zeropage-Adressierung.
FI (Operand, ANFADR ENDADR) sucht alle Befehle mit unmittelbarer Adressierung (immediate).
Beispiel: Gesucht werden Befehle, die zum Beispiel das Y-Register mit 01 laden. FI01,2000 6000 findet LDY #01 in Adresse $4000.
Sie sehen, SMON bietet eine Fülle von verschiedensten FIND-Routinen, mit denen alles gesucht und auch gefunden (!) werden kann. Zum Üben wollen wir ein großes Preisausschreiben veranstalten, bei dem Sie zumindest an Erfahrung gewinnen können! Hier sind die Aufgaben, die es zu lösen gilt:
1. Wie oft wird von SMON aus in das Betriebssystem gesprungen ($E000 — $FFFF)?
2. Welche Zeropage-Adressen benutzt SMON?
3. Wo wird die Hintergrundfarbe ($D021), wo die Schreibfarbe ($0286) gesetzt?
4. Wo sind im SMON die Tabellen untergebracht?
5. An einigen Stellen stehen Befehle, die die Register unmittelbar mit dem Highbyte des SMON-Speicherbereichs laden (dez. 49152 — 52208). Rechnen Sie die HEX-Werte aus und suchen Sie die Speicherstellen.
6. An zwei Stellen stehen aufeinanderfolgend fünf Nullen. Wo?
Notieren Sie Ihre Lösung bitte auf einem Zettel und werfen Sie diesen fort. Der Rechtsweg ist ausgeschlossen...

Der Lösung ein Stück näher...

Auch heute werden Sie nicht entlassen, ohne daß wir Ihnen einige Tips für eigene Assemblerprogramme mitgeben. Erinnern Sie sich noch an den in der letzten Ausgabe angesprochenen 16-Bit-Vergleich? Dieser wird im SMON zum Beispiel dazu benutzt, festzustellen, ob ein Programmteil weiter durchlaufen werden soll, oder ob das Ende erreicht ist. Das prüft SMON bei fast allen Befehlen, jüngstes Beispiel sind die FIND-Kommandos. Zur Erinnerung: Wir wollten zwei 16-Bit-Zahlen vergleichen, der Prozessor kann aber nur mit 8-Bit-Zahlen umgehen.
Wir brauchen dazu einen hochlaufenden Zähler (er heißt in unserem Beispiel »Programmzähler« PC und besteht aus Highbyte PCH und Lowbyte PCL) und einen End-Zeiger (ENDHI und ENDLO). Unser Programm dafür sah folgendermaßen aus:

```
LDA    PCL
CMP    ENDLO
LDA    PCH
SBC    ENDHI
```

Anschließend haben wir das Carry-Flag überprüft und festgestellt, daß schon bei Übereinstimmung beider Adressen dieses Flag gesetzt war. In unserem Falle würde also ein Beenden des Programmteils mit »BCS ENDE« zu einer »Unterschlagung« des letzten noch auszuführenden Befehls führen. Um hier einen Weg aus dem Dilemma zu finden, wollen wir uns das Verhalten der Zero- und Carry-Flagge im Status-Register einmal genauer ansehen. Und zwar in Abhängigkeit von PC und dem ENDE-Zeiger.
Sie sehen in Listing 1, daß wir den Programmzähler (PCL/PCH) nach $FB/$FC und den Endezeiger (ENDLO/ENDHI) nach $FD/$FE schreiben. Dann springen wir die Routine an, die die Überprüfung auf erreichtes Ende im SMON vornimmt (CMPEND in $C466). Zum Abschluß sorgt der BRK-Befehl dafür, daß wir wieder im SMON landen. Schauen Sie sich die entsprechende Routine im SMON mit »D C466« an. Sie werden erkennen, daß sie den oben angesprochenen 16-Bit-Vergleich durchführt.

Speichern Sie jetzt dieses Programm mit »S "CMP-TEST" 4100 4112« ab und starten es mit »G4100«. Nachdem das Programm gelaufen ist, meldet es sich mit der Registeranzeige zurück. Achten Sie dabei vor allem auf die Statusregister-Anzeige rechts, uns interessieren die Werte für Z und C (Zero- und Carry-Flagge).

Tippen Sie bitte einmal folgendes Programm ein (mit »A 4100«):

```
4100   LDA   #00
4102   STA   FB      (=PCL)
4104   STA   FD      (=ENDLO)
4106   LDA   #C0
4108   STA   FC      (=PCH)
410A   LDA   #C1
410C   STA   FE      (=ENDHI)
410E   JSR   C466    (=CMPEND)
4111   BRK
```

Listing 1.

Wir wollen herausfinden, was passiert, wenn der Programmzähler (PC) kleiner, gleich oder größer als ENDE ist. Wenn Sie jetzt in Speicherstelle $4106 für PCH den Wert C1 einsetzen, können Sie den Vorgang wiederholen und die Änderung der Flaggen notieren. Anschließend setzen Sie C2 für PCH in $4106 ein. Tippen Sie »D 4100 4112«, gehen mit dem Cursor in die Zeile 4106 und überschreiben den Wert #C0 mit dem neuen Wert #C1 beziehungsweise #C2.

		PC<END	PC=END	PC>END
PC	FC/FB	C0/00	C1/00	C2/00
END	FE/FD	C1/00	C1/00	C1/00
		ZC 00	ZC 11	ZC 01

So sollte Ihre Tabelle zum Schluß aussehen. Im ersten Fall (PC ist kleiner als END) ist das Carry-Flag gelöscht. Dann (PC ist gleich END) sind Z- und C-Flagge gesetzt, zum Schluß ist nur noch das C-Flag 1. Jetzt können wir unseren Vorstellungen entsprechend reagieren und mit den Branch-Befehlen verzweigen. Sehen Sie sich mit dem Disassembler auch einmal andere Routinen im SMON daraufhin an.

Hinweise zum Abtippen

Aus vielen Telefonanrufen ist uns klar geworden, daß der bisherige DATA-Lader oft die Fehlerquellen nicht genau genug aufzeigte. Mit diesem Lader dürften Sie beim Abtippen keine Schwierigkeiten mehr haben. Danach sollten Sie das Ladeprogramm auf alle Fälle auf Diskette oder Kassette abspeichern. Nach Eingabe von RUN muß der Lader bis zum READY durchlaufen. Um den neuen Teil an SMON anzukoppeln, müssen Sie jetzt den SMON vom letzten Mal mit »,8,1« laden und mit SYS 49152 starten. Jetzt können Sie mit

S "SMON $C000" C000 CBF1

das bis hier komplette Programm abspeichern. Natürlich müssen Diskettenbesitzer eine andere Diskette einlegen oder das alte SMON-Programm nach dem Laden (!) löschen. Und bis zum nächsten Mal: Üben, üben, üben!

(N. Mann/D. Weineck/gk)

Listing 2. Der dritte Teil des SMON als Basic-Lader

```
1 REM ********************************     <250>
2 REM *     ++++ SMON TEIL 3 ++++      *     <219>
3 REM *                                *     <230>
4 REM *  VON N.MANN UND D.WEINECK      *     <223>
5 REM *  FLEETRADE 40, 2800 BREMEN 1   *     <184>
6 REM *  TEL: 0421/ 493090             *     <055>
7 REM *       0421/ 231401             *     <011>
8 REM ********************************     <001>
9 :                                          <067>
10 DIM H(75) : FOR I=0 TO 9                  <088>
20 H(48+I)=I : H(65+I)=I+10 : NEXT           <250>
30 FOR I=51895 TO 52208:READ A$              <113>
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1))    <063>
50 D=H(H)*16+H(L) : S=S+D : POKE I,D         <181>
60 A=A+1:IF A< 8 THEN NEXT : A=-1            <110>
65 PRINT "ZEILE:";1000+Z;                    <012>
70 READ V : Z=Z+1 : IF V=S THEN 85           <210>
80 PRINT"PRUEFSUMMENFEHLER!";999+Z:STOP      <015>
85 IF A<0 THEN END                           <043>
90 S=0 : A=0 : PRINT : NEXT : END            <053>
98 :                                         <156>
99 :                                         <157>
1000 DATA 20,64,C2,A2,27,20,40,C3, 818       <200>
1001 DATA 20,23,C3,A0,08,A2,00,20, 624       <182>
1002 DATA 4C,C3,A1,FB,20,39,C4,D0, 1176      <063>
1003 DATA F9,A2,00,20,5D,C4,F0,03, 975       <243>
1004 DATA 4C,BA,CA,60,20,7E,C2,A0, 1072      <068>
1005 DATA 03,20,CF,FF,88,D0,FA,20, 1123      <065>
1006 DATA CA,C2,C9,2E,F0,02,91,FB, 1281      <086>
1007 DATA C8,C0,20,90,F2,60,20,7A, 1060      <013>
1008 DATA C2,A2,00,A1,FB,C1,FD,D0, 1422      <083>
1009 DATA 0B,20,67,C3,E6,FD,D0,F3, 1275      <079>
1010 DATA E6,FE,D0,EF,20,4C,C3,4C, 1310      <104>
1011 DATA 23,C3,A9,FF,A2,04,95,FA, 1219      <083>
1012 DATA CA,D0,FB,20,CA,C2,A2,05, 1256      <091>
1013 DATA DD,6E,C0,F0,45,CA,D0,F8, 1490      <116>
1014 DATA 86,A9,20,B4,CB,E8,20,CF, 1189      <092>
1015 DATA FF,C9,20,F0,F3,C9,2C,D0, 1424      <102>
1016 DATA 03,20,7A,C2,20,51,C3,A4, 823       <224>
1017 DATA A9,B1,FB,20,D6,CB,D0,18, 1278      <100>
1018 DATA 88,10,F6,20,23,C3,20,4C, 768       <232>
1019 DATA C3,A4,D3,C0,24,90,09,20, 983       <244>
1020 DATA 94,C4,20,72,C4,20,51,C3, 994       <236>
1021 DATA 20,63,C4,90,DA,A0,27,4C, 964       <007>
1022 DATA 96,C4,BD,73,C0,85,A8,BD, 1332      <098>
1023 DATA 78,C0,85,A9,AA,F0,06,20, 1062      <055>
1024 DATA B4,CB,CA,D0,FA,20,7A,C2, 1391      <126>
1025 DATA 20,CB,C4,20,2C,C5,A5,A8, 1037      <079>
1026 DATA 24,AB,D0,09,A8,D0,21,A5, 998       <031>
1027 DATA AD,D0,1D,F0,0D,A4,A9,B9, 1181      <119>
1028 DATA AD,00,20,D6,CB,D0,11,88, 983       <027>
1029 DATA D0,F5,B4,AA,20,8C,C5,20, 1156      <077>
1030 DATA 6F,C4,20,66,C4,90,D1,60, 1086      <056>
1031 DATA 20,6A,C6,F0,F5,20,C0,CB, 1248      <090>
1032 DATA 9D,CC,03,BD,3C,03,9D,6C, 881       <075>
1033 DATA 03,20,CA,C2,A0,0F,C9,2A, 849       <041>
1034 DATA D0,02,A0,00,20,AF,C2,9D, 928       <022>
1035 DATA 3C,03,98,9D,9C,03,60,85, 760       <008>
1036 DATA B4,4A,4A,4A,4A,59,6C,03, 676       <042>
1037 DATA 39,CC,03,29,0F,D0,0A,A5, 703       <031>
1038 DATA B4,59,3C,03,39,9C,03,29, 589       <014>
1039 DATA 0F,60, 111                         <093>
```

Checksummer — keine Fehler mehr beim Abtippen von Listings

Das Programm Checksummer 64 ist für all die Leute gedacht, die sich manchmal vor Verzweifelung »die Haare raufen« könnten. Da sitzt man mehrere Stunden, um ein gutes Programm aus dem 64'er-Magazin abzutippen, und dann: ein Fehler in der DATA-Zeile oder ein falscher Buchstabe... und schon geht die Fehlersuche los. Hier soll der Checksummer 64 weiterhelfen.

Der Checksummer 64 ist ein kleines Maschinenprogramm, das, wenn es aktiviert ist, Sie sofort davon unterrichtet, ob Sie die jeweilige Programmzeile korrekt eingegeben haben.

1. Tippen Sie den Basic-Lader sorgfältig ein. Es gibt zwei Versionen: eine für den Commodore 64 und eine für den VC 20.

2. Bevor Sie »RUN« eingeben, speichern Sie den Basic-Lader bitte erst ab, denn wenn Sie zum Beispiel einen Fehler bei den eingetippten POKE-Anweisungen gemacht haben, ist es möglich, daß der Rechner aussteigt. Heben Sie sich den abgespeicherten Checksummer 64 auf — Sie werden ihn immer wieder brauchen, wenn Sie ein Basic-Programm aus dem 64'er eintippen wollen.

3. Der Checksummer 64 überprüft sich selbst. Wenn Sie einen Fehler in den DATAs gemacht haben, listen Sie die fehlerhafte Zeile einfach, korrigieren sie und starten dann das Programm neu.

4. Nach Initialisierung des Maschinenprogramms ist der Checksummer 64 aktiviert. Er steht innerhalb des Betriebssystems und verbraucht kein einziges Byte Speicherplatz. Es sei hier für Interessierte gesagt, daß selbst alle Sprungvektoren unverändert bleiben, das Programm also mit einer Vielzahl von anderen Programmier-Spracherweiterungen wie etwa Exbasic Level II problemlos zusammenarbeitet. Achten Sie aber darauf, daß bestimmte Spracherweiterungen das hinter dem ROM liegende RAM für Hires-Grafiken benutzen. Wird zum Beispiel eine Hires-Grafik von Simons Basic aus angesprochen, so wird der Checksummer 64 zerstört.

5. Wenn Sie den Checksummer 64 zwischenzeitlich nicht benutzen, können Sie ihn jederzeit mit »POKE 1, 55« desaktivieren. Auch durch Drücken der Run-Stop- und der Restore-Taste wird der Checksummer 64 desaktiviert. Wollen Sie, daß der Checksummer 64 auch noch nach Drücken dieser Tastenkombination erhalten bleibt, so geben Sie bei aktiviertem Checksummer 64 »POKE 64982, 53« ein. Der Checksummer 64 ist dann nur durch »POKE 1, 55« abschaltbar.

Wollen Sie den Checksummer 64 wieder einschalten, so geben Sie bitte »POKE 1, 53« ein.

Das Maschinenprogramm bleibt solange erhalten, bis der Computer ausgeschaltet, oder wenn von anderen Programmen auf das hinter dem ROM liegende RAM zugegriffen wird.

6. Eine Checksumme wird nur dann ausgegeben, wenn der Commodore 64 (VC 20) eindeutig erkennt, daß Sie eine Zeile bestehend aus der Zeilennummer und zumindestens einem alphanumerischen Zeichen eingegeben haben. Ansonsten reagiert der Commodore 64 normal.

Hinweis: Wenn Sie bei aktiviertem Checksummer 64 ein Programm mit »LOAD« in den Speicher holen, wird auch eine Checksumme ausgegeben. Dies liegt jedoch an rechnerinternen Routinen und hat keine weitere Bedeutung, stellt insbesondere keine Gefahr für das geladene Programm dar, da alle Pointer richtig gesetzt werden.

Nach Eingabe von RUN wird zunächst einmal das ROM in das RAM des Commodore 64 verschoben, wonach der Basic-Interpreter modifiziert wird. Dadurch hat man den Vorteil, trotz einer zusätzlichen Routine das gesamte RAM des Rechners zur Verfügung zu haben. Nach ordnungsgemässem Ablauf des Programms können Sie sofort mit Eingaben beginnen. Für Maschinensprache-Spezialisten weise ich darauf hin, daß ich ausnutze, daß die Einschaltmeldungen des Rechners nur nach einem Reset generiert wird. Der Textbereich, in dem die Meldung steht, wird von dem erzeugten Maschinenprogramm überschrieben.

Alle veröffentlichten Listings sind mit einer Checksumme versehen, die am Ende jeder Programmzeile steht. Diese Checksumme steht zwischen < und >. Sie wird beim Eintippen des Programms nicht mit eingegeben. Die Zahl zwischen den beiden Zeichen stellt lediglich eine Information für Sie dar. Wenn Sie diese Checksumme dennoch mit eintippen, werden Sie schnell bemerken, daß Sie etwas falsch gemacht haben. Bei aktiviertem Checksummer 64 wird nämlich nach Eingabe einer Basic-Zeile, die mit Return beendet wird, in die linke obere Bildschirmecke die Checksumme eingeblendet, die mit der Summe aus dem veröffentlichten Listing übereinstimmen muß. Ist das nicht der Fall, haben Sie die Zeile anders eingegeben, als sie im Listing dargestellt ist. Vergessen Sie also bitte nicht, daß die am Ende einer Zeile in < und > stehende Prüfsumme nicht mit eingegeben werden darf.

Der Checksummer 64 ist so ausgelegt, daß er abhängig von der Zeilennummer und dem Text der Zeile eine Checksumme ausgibt. Beim Bilden dieser Checksumme werden Spaces (Leertaste) überlesen, was für Sie bedeutet, das es egal ist, wieviel Leerzeichen Sie zwischen den Worten lassen, da Sie für den Programmablauf ohnehin keine Bedeutung haben. Aber manchmal ist das richtige Setzen von Leerzeichen doch wichtig, besonders innerhalb von Strings (Zeichenketten), die gedruckt werden sollen. Seien Sie deshalb besonders genau bei Leerzeichen, die innerhalb von Anführungszeichen stehen, denn meistens ermöglichen nur die richtig gesetzten Spaces eine sinnvolle Textausgabe auf dem Bildschirm.

Beachten Sie auch, daß es durchaus erlaubt ist, Abkürzungen für die Commodore-Befehlswörter zu verwenden. So führt die Eingabe von »?« als Kurzschreibweise für »PRINT« nicht etwa zu einem Checksummen-Fehler, sondern wird korrekt verarbeitet und dementsprechend die Checksumme generiert. Nachdem Sie ein Listing eingegeben haben, sollten Sie es aus Sicherheitsgründen vor dem Starten abspeichern. Sie brauchen hierfür jedoch nicht den Checksummer 64 zu desaktivieren.

Hinweise zum Lesen von Listings

Die Listings haben sich ein wenig im Ausdruckformat verändert, um Ihnen das Eingeben von Programmen wesentlich zu erleichtern.

— Cursorsteuerzeichen und andere Steuerzeichen, die schwer zu lesen sind, werden von nun an in Klartext in speziellen Klammern gesetzt.

Tritt mehrmals hintereinander dasselbe Steuerzeichen auf, so wird diese Steuerzeichen-Sequenz zusammengefaßt, indem zuerst das Steuerzeichen und dann die Anzahl der Wiederholungen dieses Steuerzeichens in Klartext ausgegeben wird.

— alle Commodore-Grafikzeichen, die über Shift zu erreichen sind, werden nicht mehr als Grafikzeichen, sondern als Klartextzeichen dargestellt. Dabei wird aus dem Zeichen, das Sie auf dem Bildschirm sehen, wenn Sie die Tastenkombination Shift und »A« ansprechen, wieder ein »A«. Um dieses »A« vom normalen »A« unterscheiden zu können, ist es etwas kleiner als das gewöhnliche »A« und ist außerdem mit einem Unterstreichungszeichen versehen. Diese Vereinbarung gilt auch für sämtliche andere Commodore-Grafikzeichen, die über Shift zu erreichen sind.

— entsprechendes gilt für sämtliche Commodore-Grafikzeichen, die über die Commodore-Taste zu erreichen sind. Hier wird jedoch das jeweilige Klartextzeichen nicht unterstrichen, sondern überstrichen.

Erläuterungen zu den Cursorsteuerzeichen

Cursorsteuerzeichen werden, wie schon oben erwähnt, umdefiniert. Sie sehen hier eine Liste der möglichen Ausdrücke, die für ein Cursorsteuerzeichen im Listing auftauchen können. Gleichzeitig ersehen Sie aus der Tabelle, welche Taste beziehungsweise Tastenkombination zu drücken ist, damit dieses Steuerzeichen richtig in Ihr Programm übernommen wird. Beachten Sie, daß Sie die Steuercodes nur dann als reverses Zeichen sehen können, wenn der Rechner im »Quote-Modus« arbeitet, das heißt, daß er sich im Gänsefüßchenmodus befindet.

Wenn Sie lesen	drücken Sie
CTRL steht für Control-Taste, so bedeutet[CTRL-A], daß Sie die Control-Taste und die Taste »A« drücken müssen. Im folgenden steht:	
[down]	Taste neben rechtem Shift, Cursor unten
[up]	Shift-Taste & Taste neben rechtem Shift, Cursor hoch
[clear]	Shift-Taste & 2. Taste ganz rechts oben
[inst]	Shift-Taste & Taste ganz rechts oben
[home]	2. Taste von ganz rechts oben
[del]	Taste ganz rechts oben
[right]	Taste ganz rechts unten
[left]	Shift-Taste & Taste unten rechts
[space]	Leertaste
[f1]	grauer Tastenblock rechts
[f3]	grauer Tastenblock rechts
[f5]	grauer Tastenblock rechts
[f7]	grauer Tastenblock rechts
[f2]	grauer Tastenblock rechts & Shift
[f4]	grauer Tastenblock rechts & Shift
[f6]	grauer Tastenblock rechts & Shift
[f8]	grauer Tastenblock rechts & Shift
[return]	Shift-Taste & Return
[black]	Control-Taste & 1
[white]	Control-Taste & 2
[red]	Control-Taste & 3
[cyan]	Control-Taste & 4
[purple]	Control-Taste & 5
[green]	Control-Taste & 6
[blue]	Control-Taste & 7
[yellow]	Control-Taste & 8
[rvson]	Control-Taste & 9
[rvoff]	Control-Taste & 0
[orange]	Commodore-Taste & 1
[brown]	Commodore-Taste & 2
[lig.red]	Commodore-Taste & 3
[grey 1]	Commodore-Taste & 4
[grey 2]	Commodore-Taste & 5
[lig.green]	Commodore-Taste & 6
[lig.blue]	Commodore-Taste & 7
[grey 3]	Commodore-Taste & 8

Wenn Sie sich erst einmal an die in Klartext geschriebenen Steuerzeichen gewöhnt haben, werden Sie den Vorteil dieser Schreibweise erkennen. Der zu dem jeweiligen Steuerzeichen gehörende Klartext ist so verfaßt, daß Sie leicht die Taste beziehungsweise die Tastenkombination finden, die Sie drücken müssen.

Checksummer VC 20

Der Checksummer VC 20 ist im Prinzip genauso aufgebaut wie der Checksummer 64. Da beim VC 20 jedoch nicht die Möglichkeit besteht, das ROM softwaremäßig zu modifizieren, mußte ein anderer Weg als beim Commodore 64 gewählt werden, um die Checksumme zu generieren.

In ihrer Funktionsweise unterscheiden sich der Checksummer VC 20 und der Checksummer 64 nicht. Es gelten folgende Sonderregelungen bei der Benutzung des Checksummer VC 20:

— da der Basic-Bereich nicht belegt werden soll, ist das Programm im Kassettenpuffer abgelegt.

— angeschaltet wird der Checksummer VC 20 mit »SYS 955«

— Abschaltung des Checksummer VC 20 wird mit »SYS 58459« vollzogen

ACHTUNG: Nehmen Sie keine Kassetten-Operationen vor, wenn der Checksummer VC 20 eingeschaltet ist. Da das Betriebssystem den Kassettenpuffer mit Daten belegt, kann der Checksummer VC 20 überschrieben werden, was zur Folge hat, daß sich der Rechner bei aktiviertem Checksummer VC 20 »aufhängt«. Wollen Sie deshalb ein Programm auf (von) Kassette abspeichern (laden), so müssen Sie erst den Checksummer VC 20 abschalten (SYS 58459).

Daraufhin kann der Kassettenpuffer mit Daten überschrieben werden, ohne daß der Rechner »aussteigt«.

Als Sicherung wird bei der Initialisierung geprüft, ob das zuletzt angesprochene Peripherie-Gerät der Kassettenrecorder war. Ist das der Fall, so werden die Betriebssystemroutinen LOAD und SAVE für die Benutzung gesperrt. Der Rechner meldet bei Aufruf einer dieser beiden Routinen READY, ohne weitere Aktionen durchzuführen. Diese Sicherung kann man nach

der Tipparbeit aufheben, wenn man den Checksummer VC 20 mit SYS 58459 abschaltet. Dadurch wird der Kassettenpuffer für andere Daten freigemacht. Weiterhin wird dann durch gleichzeitiges Drücken der Tasten »Run-Stop & Restore« erreicht, daß die Betriebssystemroutinen LOAD und SAVE wieder eingerichtet werden.

— Bei Benutzung einer Diskettenstation brauchen Sie nicht darauf zu achten, daß bei LOAD beziehungsweise SAVE der Checksummer VC 20 überschrieben wird, da der Kassettenpuffer für die Diskettenstation normalerweise nicht genutzt wird. Deshalb können Sie die beiden Routinen weiterhin normal nutzen, sofern der Rechner bei der Initialisierung des Checksummer VC 20 feststellt, daß das zuletzt angesprochene Peripherie-Gerät nicht der Kassettenrecorder war.

— bedingt durch den anderen Aufbau des Checksummer VC 20 wird anders als beim Checksummer 64 nach der LOAD-Routine keine Checksumme ausgegeben.

— wird eine Zeile gelöscht, also eine Zahl zwischen 0 und 63999 eingegeben, und danach Return gedrückt, so wird eine Checksumme ausgegeben, die aber keine Bedeutung hat.

Viel Spaß beim Eintippen von Programmen mit dem neuen Checksummer!

(F. Lonczewski / gk)

```
10 REM ********************************     <175>
20 REM *                              *     <247>
30 REM *       CHECKSUMMER 64         *     <162>
40 REM *                              *     <011>
50 REM *           64'ER              *     <061>
60 REM *                              *     <031>
70 REM *        COMMODORE 64          *     <056>
80 REM *                              *     <051>
90 REM ********************************     <255>
100 PRINT"[CLEAR,SPACE13,RVSON]CHECKSUMMER
   [SPACE]64[RVOFF]"                        <025>
110 PRINT                                   <007>
120 PRINT"[DOWN2,SPACE4]ICH[SPACE]ARBEITE!
   [SPACE]BITTE[SPACE]ETWAS[SPACE]GEDULD." <071>
130 FOR I=40960 TO 49151:POKE I,PEEK(I):NEXT
                      <007>
140 FOR I=57344 TO 65535:POKE I,PEEK(I):NEXT
                      <025>
150 POKE 1,53:POKE 42289,96:POKE 42290,228 <001>
160 FOR I=58464 TO 58554:GOSUB 220:POKE I,A
                      <184>
170 PS=PS+A+1:NEXT I                        <109>
180 IF PS<>11187 THEN PRINT"PRUEFSUMMENFEHLER
   [SPACE]!":END                            <180>
190 PRINT"[DOWN4,SPACE9]CHECKSUMMER[SPACE]
   AKTIVIERT."                              <247>
200 PRINT"[DOWN2]AUSSCHALTEN[SPACE]
   :[SPACE]POKE1,55"                        <050>
210 PRINT"[DOWN]ANSCHALTEN[SPACE2]
   :[SPACE]POKE1,53":NEW                    <171>
220 READ A$:IF LEN(A$)<>2 THEN PRINT"TIPPFEHLER
   [SPACE]IN[SPACE]ZEILE"PEEK(63)+PEEK(64)*256
   :END                                     <201>
230 A1=ASC(A$):A2=ASC(RIGHT$(A$,1))         <216>
240 IF A1<48 OR A1>57 THEN IF A1<65 OR A1>70 TH
   EN 310                                   <130>
250 IF A2<48 OR A2>57 THEN IF A2<65 OR A2>70 TH
   EN 310                                   <144>
260 IF A1>64 THEN A1=A1-55:GOTO 280         <204>
270 IF A1<58 THEN A1=A1-48                  <128>
280 IF A2>64 THEN A2=A2-55:GOTO 300         <220>
290 IF A2<58 THEN A2=A2-48                  <151>
300 A=A1*16+A2:RETURN                       <138>
310 PRINT"UNGUELTIGER[SPACE]HEXCODE[SPACE]IN
   [SPACE]ZEILE"PEEK(63)+PEEK(64)*256:END  <021>
320 DATA A0,02,A9,00,85,02,B1,5F            <098>
330 DATA F0,0F,C9,20,D0,03,C8,D0            <146>
340 DATA F5,18,65,02,85,02,4C,6E            <126>
350 DATA E4,C0,04,30,F1,C6,D6,A5            <169>
360 DATA D6,48,A2,03,A9,20,9D,01            <150>
370 DATA 04,BD,B7,E4,20,D2,FF,CA            <238>
380 DATA 10,F2,A6,02,A9,00,20,CD            <169>
390 DATA BD,A9,3E,20,D2,FF,68,85            <245>
400 DATA D6,20,6C,E5,A9,8D,20,D2            <229>
410 DATA FF,4C,80,A4,5C,48,20,C9            <244>
420 DATA FF,AA,68,90,01,8A,60,09            <234>
430 DATA 3C,12,13                           <199>
```

Der Checksummer für den Commodore 64

```
10 REM****************                      <057>
20 REM*              *                      <247>
30 REM* CHECKSUMMER  *                      <056>
40 REM* VERSION VC20 *                      <044>
50 REM*              *                      <021>
60 REM****************                      <107>
70 PRINT"[CLEAR,SPACE2,RVSON]CHECKSUMMER[SPACE2]
   VC-20[RVOFF]"                            <185>
80 PRINT                                    <233>
90 PRINT"[DOWN]EINEN[SPACE]MOMENT,[SPACE]
   BITTE..."                                <181>
100 FOR I=827 TO 993:GOSUB 180:POKE I,A     <177>
110 PS=PS+A+1:NEXT I                        <049>
120 IF PS<>20612 THEN PRINT"[DOWN]
   PRUEFSUMMENFEHLER[SPACE]!":END           <130>
130 SYS 955:PRINT"CHECKSUMMER[SPACE]AKTIVIERT."
                    <242>
140 PRINT"AN[SPACE]:SYS955"                 <212>
150 PRINT"[DOWN]AUS:SYS58459,[SPACE]BEI[SPACE]
   CAS-[SPACE4]SETTE[SPACE]ZUSAETZLICH[SPACE5]
   RUN/STOP[SPACE]&[SPACE]RESTORE"          <068>
160 PRINT"[DOWN]BEI[SPACE]AKTIVIERTEM[SPACE]
   CHECK-SUMMER[SPACE]KEIN";                <105>
170 PRINT"[SPACE]CASSETTEN-BETRIEB[SPACE](LOAD,
   [SPACE]SAVE)[SPACE2]ERLAUBT!":NEW        <051>
180 READ A$:IF LEN(A$)<>2 THEN PRINT"TIPPFEHLER
   [SPACE]IN[SPACE]ZEILE"PEEK(63)+PEEK(64)*256
   :END                                     <161>
190 A1=ASC(A$):A2=ASC(RIGHT$(A$,1))         <176>
200 IF A1<48 OR A1>57 THEN IF A1<65 OR A1>70 TH
   EN 270                                   <095>
210 IF A2<48 OR A2>57 THEN IF A2<65 OR A2>70 TH
   EN 270                                   <109>
220 IF A1>64 THEN A1=A1-55:GOTO 240         <159>
230 IF A1<58 THEN A1=A1-48                  <087>
240 IF A2>64 THEN A2=A2-55:GOTO 260         <184>
250 IF A2<58 THEN A2=A2-48                  <110>
260 A=A1*16+A2:RETURN                       <098>
270 PRINT"UNGUELTIGER[SPACE]HEXCODE[SPACE]IN
   [SPACE]ZEILE"PEEK(63)+PEEK(64)*256:END  <237>
280 DATA 20,5F,03,86,7A,84,7B,20            <061>
290 DATA 73,00,AA,F0,F3,A2,FF,86            <130>
300 DATA 3A,90,0A,A2,00,86,FF,20            <097>
310 DATA 79,C5,4C,E1,C7,A2,01,86            <127>
320 DATA FF,4C,9C,C4,A6,FF,E0,01            <199>
330 DATA F0,03,4C,60,C5,A0,02,A9            <125>
340 DATA 00,85,FE,B1,5F,F0,0F,C9            <186>
350 DATA 20,D0,03,C8,D0,F5,18,65            <141>
360 DATA FE,85,FE,4C,76,03,C0,04            <193>
370 DATA 30,F1,C6,D6,A5,D6,48,A2            <198>
380 DATA 03,A9,20,9D,01,04,BD,B7            <180>
390 DATA 03,20,D2,FF,CA,10,F2,A6            <217>
400 DATA FE,A9,00,20,CD,DD,A9,3E            <016>
410 DATA 20,D2,FF,68,85,D6,20,87            <220>
420 DATA E5,A9,8D,20,D2,FF,A2,00            <003>
430 DATA 86,FF,F0,AE,09,3C,12,13            <002>
440 DATA A9,3B,8D,02,03,A9,03,8D            <249>
450 DATA 03,03,A5,BA,C9,01,D0,10            <235>
460 DATA A9,74,8D,30,03,8D,32,03            <239>
470 DATA A9,C4,8D,31,03,8D,33,03            <007>
480 DATA AD,88,02,8D,90,03,60               <113>
```

Der Checksummer für den VC 20

Achtung !! VC 20/64

SPEEDDOS 64

Die Hardwareerweiterung der Superlative

Macht Ihre Floppy bis zu **10 x** so schnell

Bietet über **100** neue Möglichkeiten!

Beispiel: Ladezeit bisher **1** Minute
mit SPEEDDOS nur **6** Sekunden



Die Superhardware zum Commodore 64

DER HELLE WAHNSINN !!

Und das bietet Ihnen SPEEDDOS 64

10 x schneller als bisher können sie mit Speeddos 64 Ihre Programme in den Rechner laden. Also eine enorme Zeitersparnis...

8 x schneller werden die Daten zwischen der Floppy und dem Rechner übertragen! Somit wird bei allen Diskettenoperationen — Dateien u. sonst. — eine Geschwindigkeitssteigerung zwischen 2- bis 5 mal erreicht.

Formatieren können Sie Ihre Disketten doppelt so schnell wie bisher.

98 % aller Programme laufen mit Speeddos 64, da dieses Programm keinerlei RAM Speicherplatz verbraucht! Ebenfalls brauchen zum Betrieb von Speeddos keine Zusatzprogramme eingeladen werden.

Laden und Starten des ersten Programmes von Diskette mit nur einem Tastendruck! Einlesen des Directorys ohne Programmverlust! Einladen von Programmen mittels der Cursortaste (einfach auf den Namen fahren)

Eine Centronics-Schnittstelle ist softwaremäßig gleich mitintegriert.

Die Funktionstasten — abschaltbar — sind mit vielen Befehlen belegt.

Verbesserungen wurden nicht nur im allgemeinen durchgeführt, sondern auch im Detail sinnvoll durchdacht. Bessere Zentrierung des Laufwerkes durch gezielte Anlaufsteuerung des Motors, und...und...und...

Tool's gehören zum guten Ton dieser Erweiterung! Nicht nur umfangreiche Diskettenhilfen sind integriert, sondern auch anspruchsvolle Programmierhilfen.

Das haut den stärksten Mann vom Hocker!



Datenkassetten: C 10 nur 0,99 DM
Plastikbox verschweißt C 20 nur 1,19 DM

Datarecorder mit Mithörverstärker (keine LOAD ERRORS)
Direkt an VC 20/64 anschließbar.
Nur 79,— DM!

Echt stark



Joystick mit **2** Feuerknöpfen nur 16,95 DM !!



Disketten für 1541
10er Pack nur 39,95 DM
10er Pack **farbig** 49,95 DM

Disk Dubbler nur 19,80 DM!

Diskettenbox für **80** Disketten ! Nur 39,90 DM !!

Erweiterung **16 KB** für VC 20 ! Schaltbar nur 99,— DM !

Zugreifen! Echte Knüller!

SPEEDDOS 64

Der Knüllerpreis !

Natürlich...
inkl. einer ausführlichen deutschen Einbauanleitung! Zum Superpreis von sage und schreibe nur 269,— DM ! (Kompletter Einbausatz! Inkl. Umrüstanleitung! Zeitaufwand ca. 10 Minuten)



Sketch Pad 64

Das Grafiktablett zum Commodore 64! Software im **Steckmodul! Keine** Diskettenstation erforderlich! Echt **erstklassig!** Befehle zum Linienziehen/Rechtecke/Kreise/Füllen/Zoom Funktion/Verschiedene Pinselstärken/Expert Funktion/**Direktes Abzeichnen von einer A4-Vorlage**. Einsetzbereit im kommerziellen und Homebereich. Natürlich inkl. **Deutsche Anleitung** nur 239,— DM !

Heute noch bestellen !!



COUPON

- ○ Hiermit bestelle ich______ Exemplare SPEDDOS für den Commodore C 64 zum Preis von je 269,— DM
- ○ Hiermit bestelle ich den folgenden Artikel

____

____

- ○ Ich bezahle per Nachnahme ohne Nachnahmekosten
- ○ Ich bezahle per Scheck (liegt anbei)

Name ______________

Straße ______________

Ort ______________

Mein Computer ______________

Mindestbestellwert 20,— DM

Telefonservice Täglich von 10–18 Uhr ! 02367/446

Keine Nachnahmekosten !

Achtung! Anzeige auf Seite 73 beachten!

S+S Soft

J. Schlüter, Schöttelkamp 23a
4620 Castrop-Rauxel 9
Tel. 02367/446

Vier Pseudo-VICs mit 32 Sprites

Sie wollen mit 32 Sprites und vier Bildschirmbereichen gleichzeitig arbeiten? Nichts leichter als das. Mit Provic 64 können simultan Grafik, Text oder veränderte Zeichensätze dargestellt werden.

Die Autoren (Jürgen, 21, Physikstudent, und Stefan, 18, Schüler) haben sich Mitte 1983 einen Commodore 64 angeschafft. Schon nach kurzer Zeit stellte sich der bei den C 64-Fans übliche Frust über die schlechte Dokumentation und die schwierige Informationsbeschaffung ein, besonders wenn es um die speziellen Grafikfähigkeiten dieses Computers geht. So sitzen wir oft stundenlang vor dem Bildschirm, der nur undefinierbare Zeichen zeigt, weil wir bei dem Versuch, die Geheimnisse des C 64 zu enträtseln, in irgendeinen unbekannten Darstellungsmodus geraten sind.

Dabei entdeckten wir, daß der C 64 nicht nur acht, sondern auch 16, 24, 32 oder noch mehr Sprites gleichzeitig zeigen kann. Zusätzlich ergibt sich die Möglichkeit, mehrere Bildschirmmodi zu mischen.

Nun haben wir uns entschlossen, den Kunstgriff, der dies ermöglicht, anderen C 64-Fans nicht vorzuenthalten. Also entwickelten wir ein Programm in Maschinensprache und dazu ein kleines Demonstrationsprogramm in Basic.

Zum Programm

Durch Aufruf der Initialisierungsroutine wird der Interruptmechanismus des C 64 verändert. Nicht mehr der Timer der CIA 1, sondern der VIC 6567 löst jetzt den Interrupt aus, und zwar synchron zum Takt des Bildschirmsignals. Außerdem werden vier sogenannte Pseudo-VICs eingerichtet. Alle POKEs, von Spritebewegung über Bildschirmfarbe bis zur Grafikkonfiguration, müssen ab jetzt in diese Pseudo-VICs erfolgen. Jeder dieser Pseudo-VICs ist für einen der vier Bildschirmbereiche zuständig:

Der Bildschirm wird in vier waagerechte Bereiche aufgeteilt, deren Grenzlinien fast beliebig nach oben oder unten verschoben werden können. Jeder einzelne Bereich kann acht Sprites darstellen und eine eigene Farb- und Grafikkonfiguration aufweisen. So können zum Beispiel Normalschrift, HiRes-Grafik, Multicolor-Grafik und eventuell ein selbstdefinierter Zeichensatz gleichzeitig auf dem Bildschirm gezeigt werden.

Provic 64 kann selbstverständlich wieder abgeschaltet werden (bei Kassetten- oder Diskettenoperationen nötig).

Für Maschinensprachefreaks nun eine kurze Funktionsbeschreibung der Interruptroutine:

Bei Aufruf der Einschaltroutine (SYS 52544) wird der IRQ-Vektor auf die Hauptroutine des Provic 64 gestellt und der bisherige Interrupt durch den Timer A der CIA 1 verboten, während der Raster-IRQ des VIC 6567 erlaubt wird.

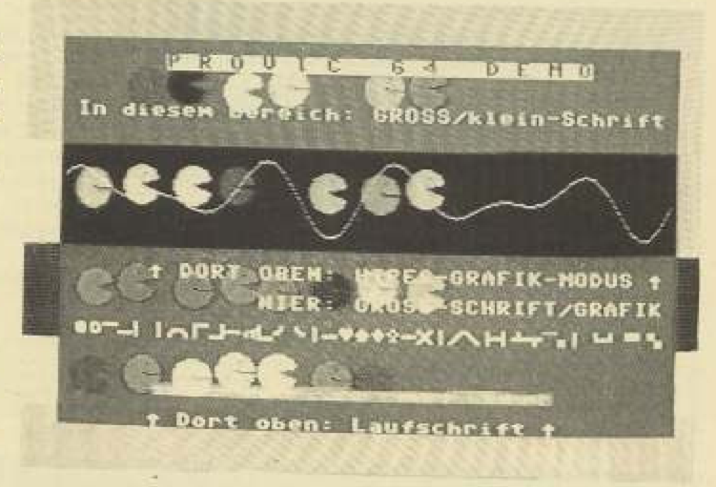
Sobald der Bildschirmstrahl die eingestellte Rasterzeile erreicht, wird ein Interrupt ausgelöst und der Prozessor bearbeitet die Hauptroutine des Provic 64. In dieser wird zunächst anhand eines Zählers ($ CFFF) festgestellt, welcher Bildschirmbereich an der Reihe ist. Dann wird die Rasterzeile, die das Ende dieses Bildschirms kennzeichnet, eingestellt.

Anschließend werden, falls ein entsprechendes Flag gesetzt ist, die Sprite- und andere Bildschirmparameter in den VIC 6567 übertragen. Nach dem Weiterzählen des IR-Zählers ($ CFFF) wird entweder der Interrupt beendet, oder zur IRQ-Routine des Betriebssystems gesprungen (nach jedem vierten Interrupt). So werden in der Sekunde 200 Interrupts (vier pro Fersehbild) ausgelöst und 50 mal in der Sekunde (einmal pro Bild) die normale IRQ-Routine abgearbeitet. Dadurch zählt die interne Uhr TI in 50stel Sekunden und TI$ wird unbrauchbar.

Beim Aufruf der Ausschaltroutine wird der Raster-IRQ des VIC 6567 unterbunden, der Interrupt des Timers A in CIA 1 erlaubt und der IRQ-Vektor auf die IRQ-Routine des Betriebssystems eingestellt.

Handhabung der Pseudo-VICs

Im Grunde ist jeder der vier Pseudo-VICs wie der echte VIC zu behandeln. Ausnahmen sind hier nur die Register 30 (Sprite-Sprite-Kollision) und 31 (Sprite-Hintergrund-Kollision), die sich auf den jeweils vorausgegangenen Bildschirmbereich beziehen. Die Register 19 und 20 (Lightpenkoordinaten), sowie 25 und 26 (IRQ-Flags und -maske) werden nicht behandelt, da diese Funktionen nur direkt über den VIC 6567 sinnvoll gehandhabt werden können. Außerdem hat jeder Pseudo-VIC noch zusätzliche Register für zwei Flags (REG 47 und REG 57), acht Sprite-Pointer (REG 48 bis REG 55), Videomatrix-Anfangsadresse Highbyte (REG 56) und die CIA 2, REG 0, Bits 0 und 1 (REG 58) (Adreßbits 14 und 15 des VIC 6567).



So werden die Fähigkeiten von »Provic 64« demonstriert

Die vier Basisadressen der PVICs sind:

PVIC 1	52992 ($ CF00)	= REG 0
PVIC 2	53056 ($ CF40)	= REG 0
PVIC 3	53120 ($ CF80)	= REG 0
PVIC 4	53184 ($ CFC0)	= REG 0

Da die Pseudo-VICs praktisch gleichberechtigt sind, hier die Registerbeschreibung eines Pseudo-VICs:

REG 0:	X-Koordinate des Sprite 0
REG 1:	Y-Koordinate des Sprite 0 Beachte: Die Y-Koordinaten sollten im Bereich des zugehörigen Bildschirmbereichs liegen, sonst ist der Sprite nicht zu sehen. Näheres siehe unten.
REG 2 bis 15:	Wie REG 0 und 1, aber für Sprites 1 — 7
REG 16:	MSB (höchstes Bit) der X-Koordinaten
REG 17:	Bits 0 bis 2: Y-Abstand der Zeichen vom oberen Bildrand in Rasterzeilen (Softscrolling!)
REG 17:	Bit 3: Umschaltung 24/25-Zeilendarstellung Bit 4: Bild an/aus: es hat keinen Sinn, das Bild teilweise ausschalten zu wollen, da der VIC dieses Bit nur einmal pro Fernsehbild prüft (also entweder das ganze Bild an oder aus) Bit 5: HiRes-Grafik-Modus an Bit 6: Hintergrundmehrfarb-Modus an Bit 7: Nummer der Interrupt-Rasterzeile Bit 8; es hat wenig Sinn, dieses Bit zu setzen, da so nur Rasterzeilen angesprochen werden, die unterhalb des Bildschirms liegen. Ist in irgendeinem Pseudo-VIC die 9-Bit-Zahl für die Rasterzeile größer als 311, so wird überhaupt kein IRQ mehr ausgelöst.
REG 18:	Nummer der Rasterzeile Bits 0 — 7; hier ist anzugeben, wann der nächste Interrupt ausgelöst werden soll, das heißt wo der Bildschirmbereich dieses PVICs zu Ende sein soll. Dabei sollte folgende Reihenfolge eingehalten werden: REG 18: PVIC 1 < PVIC 2 < PVIC 3 <PVIC 4 (Zyklische Vertauschungen möglich!)
REG 19 und 20:	nicht verwendet (siehe oben)
REG 21:	Sprite enable (einschalten)
REG 22:	Bits 0 bis 2: Softscrolling in X-Richtung Bit 3: Umschaltungen 38/40-Spaltendarstellung Bit 4: Multicolor-Modus ein Bit 5 bis 7: unbenutzt
REG 23:	Sprite vergrößern in Y-Richtung
REG 24:	Bits 1 bis 3: Adresse Zeichengenerator (Bits 11 bis 13) Bits 4 bis 7: Adresse Video-RAM (Bits 10 bis 13)
REG 25 und 26:	nicht verwendet (siehe oben)
REG 27:	Sprite-Priorität vor Hintergrund
REG 28:	Flags für Multicolor-Sprites
REG 29:	Sprite vergrößern in X-Richtung
REG 30:	Sprite-Sprite-Kollision
REG 31:	Sprite-Hintergrund-Kollision Achtung: Geben die Kollisionen des vorangegangenen Bildschirmbereichs an: Findet im Bereich von PVIC 3 eine Kollision statt, wird dies im PVIC 4 registriert. Kollisionen im Bereich von PVIC 4 werden im PVIC 1 registriert. Dieses Register muß gelöscht werden, um neue Kollisionen anzeigen zu können!
REG 32:	Rahmenfarbe
REG 33 bis 36:	Hintergrundfarben 0 bis 3
REG 37 und 38:	Multicolor-Sprite-Farben 0 und 1
REG 39 bis 46:	Farben für Sprites 0 bis 7
REG 47:	Flag für Spritebehandlung; nur wenn der Inhalt dieses Registers nicht Null ist, werden die Register, die etwas mit Sprites zu tun haben, vom PVIC in den VIC 6567 übertragen. Das sind REG 0 bis REG 16, REG 21, REG 23, REG 27 bis REG 31, REG 37 bis 46 sowie REG 48 bis 55. Ist der Inhalt Null, gelten für die Sprites die Werte des vorherigen PVICs, während die Kollisionen erst im nächsten PVIC, in dem REG 48 ungleich Null ist, angezeigt werden.
REG 48 bis 55:	Sprite-Pointer für Sprites 0 bis 7; Die Pointer auf die Bitmuster der Sprites werden nicht mehr in die Speicherzellen 2040 bis 2047 geschrieben, sondern in diese Register des PVICs.
REG 56:	In diesem Register muß das Highbyte der Video-RAM-Anfangsadresse plus 3 stehen; normalerweise also 4 + 3 = 7 (da der Bildschirm nach dem Einschalten des Computers bei 1024 beginnt, 1024 = $ 0400). Bei Verlegung des Video-RAMs ist also der Inhalt dieses Registers zu korrigieren.
REG 57:	Flag für Bildschirmparameter-Behandlung; nur wenn der Inhalt dieses Registers nicht Null ist, werden die REG 17, 22, 24, sowie 32 bis 36 und REG 58 in den VIC 6567 übertragen.
REG 58:	Bits 0 und 1: Adressbits 14 und 15 des VIC 6557; werden nach CIA 2 REG 0 Bits 0 und 1 übertragen. Mit diesen Bits kann Video-RAM, Charaktergenerator, Grafik-Bitmap in 16-KByte-Schritten verschoben werden. Da die Bits low-aktiv sind, sind sie beim Einschalten gesetzt (also REG 58 = 3). Bits 2 bis 7: unbenutzt, immer 0.

Übergang eines Sprites zwischen zwei Bildschirmbereichen:

Soll ein Sprite zwischen zwei Bildschirmbereichen wechseln, muß in beiden Bereichen derselbe Sprite (also zum Beispiel beidesmal Sprite 4) die gleiche Position besitzen, und zwar so lange, wie der Sprite die Trennlinie zwischen den Bereichen überdeckt. Wird dies nicht beachtet, werden die entsprechenden Sprites zerschnitten und verschoben.

Aktivieren von Provic 64: Von Basic aus mit SYS 52544 und von Maschinensprache aus mit JSR $CD40.

Ausschalten von Provic 64: Von Basic aus mit SYS 52970 und von Maschinensprache aus mit JSR $CEEA.

Der Basic-Lader:

Der Lader erzeugt Provic 64 aus den DATA-Zeilen und falls kein Prüfsummenfehler vorliegt, wird Provic 64 sofort als Maschinenprogramm auf Floppy oder Datasette (Zeile 400 entsprechend ändern!) abgespeichert. Dieses Maschinenprogramm enthält auch gleich die Standardwerte der Pseudo-VICs.

Laden von Provic 64: Im Programm am besten mit der Zeile IF PEEK (52544) > < 120 THEN LOAD "PROVIC 64", Gerätenummer, 1 die am Anfang des Basic-Programms stehen sollte.

Das Demonstrationsprogramm zeigt einige der Vorzüge von Provic 64. Es ist nur als Anregung gedacht, deshalb verzichten wir hier auf eine nähere Beschreibung.

Provic 64 ist nicht nur für Basic-Programmierer, sondern vor allem auch für Maschinensprache-Freaks gedacht, da erst durch schnelle Maschinenprogramme die Möglichkeiten von Provic 64 voll ausgeschöpft werden können.

(Jürgen und Stefan Haas/rg)

Tabellarische Übersicht zu Provic

Belegter Adreßraum:

$CD40	Einschaltroutine
$CD58	Interruptroutine
$CEEA	Ausschaltroutine
$CF00	Pseudo-VIC 1
$CF40	Pseudo-VIC 2
$CF80	Pseudo-VIC 3
$CFC0	Pseudo-VIC 4
$CFFF	Interruptzähler

Provic 64 einschalten:
in Basic: SYS 52544
in Maschinensprache: JSR $CD40
Provic 64 ausschalten:
in Basic: SYS 52970
in Maschinensprache JSR $CEEA
Benutzte RAM-Adressen:
in der Zero-Page: 187 ($BB)
188 ($BC)

Rechenzeitzuwachs bei aktiviertem Provic 64:

alle Spriteflags (REG 47) und Bildschirmparameterflags (REG 57) gelöscht:	+ 2,5%
für jedes gesetzte Spriteflag (REG 47):	zirka + 2,4%
für jedes gesetzte Bildschirmflag (REG 57):	zirka + 0,5%
alle Sprite- und Bildschirmflags gesetzt:	zirka + 15,0%

Falls der Rechner abstürzt rettet Run-Stop/Restore!
Die Zeitvariable TI zählt bei aktiviertem Provic 64 in 50stel Sekunden (statt 60stel);
TI$ wird somit unbrauchbar.
Zeiger für Interruptaussprung von PVIC 3 bis PVIC 4: $CEE5
Zeiger für Interruptaussprung von PVIC 1: $CEE8

Basic-Lader »Provic 64«

```
10 REM INTERRUPT-ROUTINE ZUR ERZEUGUNG      <118>
11 REM                                      <154>
12 REM EINES MAXIMAL VIERTEILIGEN           <159>
13 REM                                      <156>
14 REM BILDSCHIRMES UND 32 SPRITES          <140>
15 REM                                      <158>
16 REM AUF EINEM COMMODORE 64 COMPUTER      <103>
17 REM                                      <160>
18 REM       1984 BY GEBR. HAAS             <125>
19 REM                                      <162>
100 REM PRUEFSUMMEN-KONTROLLE               <049>
101 REM                                     <244>
110 RESTORE:PS=0                            <185>
120 FOR A=0 TO 511                          <087>
130 READ WERT                               <075>
140 PS=PS+WERT                              <112>
150 NEXT A                                  <089>
160 IF PS<>60913 THEN PRINT"DATA-PRUEFSUMME
   [SPACE]FALSCH[SPACE]!":END:*             <123>
199 REM                                     <086>
200 REM MASCHINENCODE UEBERTRAGEN           <060>
201 REM                                     <088>
210 RESTORE                                 <094>
220 FOR A=0 TO 447                          <195>
230 READ WERT                               <175>
240 POKE 52544+A,WERT                       <228>
250 NEXT A                                  <189>
260 FOR A=0 TO 63                           <182>
265 READ WERT                               <211>
270 FOR B=0 TO 3                            <139>
280 POKE 52992+B*64+A,WERT                  <022>
290 POKE 52992+B*64+18,95+50*B              <113>
295 NEXT B,A                                <089>
299 REM                                     <187>
300 REM PROVIC 64 ABSPEICHERN               <029>
301 REM                                     <189>
310 N$="PROVIC[SPACE]64"                    <220>
320 FOR A=1 TO LEN(N$)                      <016>
330 POKE 49151+A,ASC(MID$(N$,A,1))          <107>
340 NEXT A                                  <024>
350 POKE 183,LEN(N$)                        <068>
360 POKE 187,0:POKE 188,192                 <054>
370 POKE 185,1                              <005>
380 POKE 193,64:POKE 194,205                <121>
390 POKE 174,0:POKE 175,208                 <074>
400 POKE 186,8: REM FUER FLOPPY
   : 8                  FUER DATASETTE: 1
                  <174>
410 SYS 62957:  REM SAVE ROUTINE            <100>
420 END                                     <037>
999 REM                                     <121>
1000 REM PROVIC 64 MASCHINENCODE            <104>
1001 REM                                    <123>
1002 DATA 120,169,88,162,205,141,20,3,142,21,3,
   169,1,141,13,220,141,26,208              <170>
1003 DATA 141,255,207,88,96,169,1,141,25,208,
   172,255,207,240,15,136,240,9,136         <105>
1004 DATA 240,3,162,192,44,162,128,44,162,64,44,
   162,0,189,18,207,141,18,208              <253>
1005 DATA 189,47,207,208,3,76,141,206,173,30,
   208,29,30,207,157,30,207,173,31          <048>
1006 DATA 208,29,31,207,157,31,207,189,21,207,
   141,21,208,189,23,207,141,23             <200>
1007 DATA 208,189,29,207,141,29,208,189,0,207,
   141,0,208,189,1,207,141,1,208            <000>
1008 DATA 189,2,207,141,2,208,189,3,207,141,3,
   208,189,4,207,141,4,208,189,5            <004>
1009 DATA 207,141,5,208,189,6,207,141,6,208,189,
   7,207,141,7,208,189,8,207,141            <114>
1010 DATA 8,208,189,9,207,141,9,208,189,10,207,
   141,10,208,189,11,207,141,11             <001>
1011 DATA 208,189,12,207,141,12,208,189,13,207,
   141,13,208,189,14,207,141,14             <254>
1012 DATA 208,189,15,207,141,15,208,189,16,207,
   141,16,208,189,27,207,141,27             <019>
1013 DATA 208,189,28,207,141,28,208,189,37,207,
   141,37,208,189,38,207,141,38             <038>
1014 DATA 208,189,39,207,141,39,208,189,40,207,
   141,40,208,189,41,207,141,41             <019>
1015 DATA 208,189,42,207,141,42,208,189,43,207,
   141,43,208,189,44,207,141,44             <020>
1016 DATA 208,189,45,207,141,45,208,189,46,207,
   141,46,208,189,56,207,133,188            <094>
1017 DATA 169,248,133,187,160,0,189,48,207,145,
   187,200,189,49,207,145,187,200           <142>
1018 DATA 189,50,207,145,187,200,189,51,207,145,
   187,200,189,52,207,145,187               <245>
1019 DATA 200,189,53,207,145,187,200,189,54,207,
   145,187,200,189,55,207,145               <241>
1020 DATA 187,189,57,207,240,67,189,58,207,41,3,
   157,58,207,173,0,221,41,252              <037>
1021 DATA 29,58,207,141,0,221,189,17,207,141,17,
   208,189,22,207,141,22,208,189            <119>
1022 DATA 24,207,141,24,208,189,32,207,141,32,
   208,189,33,207,141,33,208,189            <019>
1023 DATA 34,207,141,34,208,189,35,207,141,35,
   208,189,36,207,141,36,208,224            <024>
1024 DATA 192,208,5,169,255,141,255,207,238,255,
   207,138,240,3,76,129,234,76              <043>
1025 DATA 49,234,120,169,49,162,234,141,20,3,
   142,21,3,169,129,141,13,220,169          <058>
1026 DATA 0,76,80,205,234,0,0,0,0,0,0,0,0,0,0,
   0,0,0,0,0,0,27,95,0,0,0,200              <139>
1027 DATA 0,21,121,240,0,0,0,0,0,14,6,1,2,3,4,0,
   1,2,3,4,5,6,7,8,0,0,0,0,0,0              <140>
1028 DATA 0,0,0,7,1,3,0,0,0,0,0             <094>
```

Demo-Listing »Provic 64«

```
6 REM  DIESES KURZE DEMO-PROGRAMM SOLL        <212>
8 REM  EIN PAAR DARSTELLUNGSFORMEN            <246>
10 REM ZEIGEN, WIE SIE MIT PROVIC-64          <161>
12 REM RELATIV EINFACH ERREICHBAR SIND.       <211>
14 REM        1984 BY GEBR. HAAS              <121>
19 REM                                        <162>
20 REM LADEN DER PROVIC-64 ROUTINE            <114>
21 REM                                        <164>
30 IF PEEK(52544)=120 THEN 100                <061>
40 PRINT"[DOWN2,SPACE]AUF[SPACE]WELCHEM[SPACE]
   DATENTRAEGER[SPACE]IST[SPACE]PROVIC-64" <216>
50 PRINT"[DOWN,SPACE]VERFUEGBAR[SPACE]([SPACE]
   FLOPPY[SPACE]=8[SPACE]/[SPACE]DATASETTE
   [SPACE]=1[SPACE])"                         <229>
60 INPUT"[SPACE]";A$:A=VAL(A$)                <077>
70 IF A<>1 AND A<>8 THEN 40                   <062>
80 LOAD"PROVIC[SPACE]64",A,1                  <046>
99 REM                                        <242>
100 REM PSEUDO-VIC'S INITIALISIEREN           <101>
101 REM                                       <244>
110 P1=52992:P2=53056:P3=53120:P4=53184
   :REM BASISADRESSEN DER PSEUDO-VIC'S        <192>
120 POKE P1+21,255:POKE P1+24,22:POKE P1+47,1
   :POKE P1+27,255:POKE P2+21,255             <112>
130 POKE P2+17,59:POKE P2+24,24:POKE P2+32,7
   :POKE P2+47,1:POKE P2+27,255               <244>
140 POKE P3+21,255:POKE P3+32,9:POKE P3+47,1
   :POKE P3+27,255:POKE P4+32,5               <253>
150 POKE P4+21,255:POKE P4+24,22:POKE P4+47,1
   :POKE P4+18,230:POKE P4+27,255             <155>
199 REM                                       <086>
200 REM SPRITE-DATEN UEBERTRAGEN              <251>
201 REM                                       <088>
210 RESTORE                                   <094>
220 FOR A=0 TO 126                            <189>
230 READ WERT                                 <175>
240 POKE 832+A,WERT                           <125>
250 NEXT A                                    <189>
299 REM                                       <187>
300 REM BILSCHIRM AUFBAUEN                    <160>
301 REM                                       <189>
310 PRINT"[CLEAR,WHITE,DOWN,RVSON,RIGHT7]
   P R O V I C   6 4   D E M O"               <092>
320 PRINT"[DOWN2,SPACE]IN[SPACE]DIESEM[SPACE]
   BEREICH:[SPACE]GROSS/KLEIN-SCHRIFT[DOWN]":
                      <117>
330 FOR A=0 TO 7:PRINT"[BLUE]
   RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
   [WHITE]";:NEXT A                           <013>
335 FOR A=0 TO 14:POKE 1289+A,34:NEXT A       <029>
340 PRINT"[SPACE6]↑[SPACE]DORT[SPACE]OBEN
   :[SPACE]HIRES-GRAFIK-MODUS[SPACE]↑"        <150>
350 PRINT"[DOWN,SPACE13]HIER
   :[SPACE]GROSS-SCHRIFT/GRAFIK"              <096>
360 PRINT"[DOWN,SPACE]QWERTYUIOP*£LKJHFSAZXCVBN
   MQWERTFGZXCVB"                             <187>
370 PRINT"[DOWN5,SPACE6]↑[SPACE]DORT[SPACE]OBEN
   :[SPACE]LAUFSCHRIFT[SPACE]↑[UP3]"
   :PRINT TAB(6);                             <246>
390 SYS 52544:REM PROVIC-64 ACTIVIEREN        <070>
399 REM                                       <031>
400 REM HIRES-GRAFIK ZEICHNEN                 <208>
401 REM                                       <033>
410 FOR A=10110 TO 12360:POKE A,0:NEXT A      <013>
420 FOR A=0 TO 8*πSTEP.08                     <015>
430 X=3+A/.08:Y=77-11*SIN(A)-9*COS(A/.7)      <201>
440 AV=8192+320*INT(Y/8)+(Y AND 7)+8*INT(X/8)
                 <016>
450 POKE AV,PEEK(AV) OR 2↑(7-(X AND 7))       <025>
460 NEXT A                                    <144>
470 LA$="***[SPACE]VON[SPACE]HAASI SOFT[SPACE]
   ***[SPACE2]FUER[SPACE]DAS[SPACE]64'ER[SPACE]
   MAGAZIN[SPACE3]"                           <072>
480 LA$=LA$+"***[SPACE3]P R O V I C   6[SPACE]4
   [SPACE3]"                                  <158>
490 LA$=LA$+LEFT$(LA$,25):R=53266             <074>
499 REM                                       <131>
500 REM DEMONSTRATINS-SCHLEIFE                <223>
501 REM                                       <133>
510 REM SPRITES SETZEN                        <145>
511 REM                                       <143>
520 FOR A=0 TO 7                              <137>
530 POKE P1+2*A,30+24*A+7*RND(1)
   :POKE P1+2*A+1,60+6*RND(1)                 <156>
540 POKE P1+39+A,RND(1)*16:POKE P1+48+A,
   13.5+RND(1)                                <224>
550 POKE P2+2*A,30+24*A+7*RND(1)
   :POKE P2+2*A+1,110+6*RND(1)                <222>
560 POKE P2+39+A,RND(1)*16:POKE P2+48+A,
   13.5+RND(1)                                <246>
570 POKE P3+2*A,30+24*A+7*RND(1)
   :POKE P3+2*A+1,160+6*RND(1)                <249>
580 POKE P3+39+A,RND(1)*16:POKE P3+48+A,
   13.5+RND(1)                                <012>
590 POKE P4+2*A,30+24*A+7*RND(1)
   :POKE P4+2*A+1,207+6*RND(1)                <017>
600 POKE P4+39+A,RND(1)*16:POKE P4+48+A,
   13.5+RND(1)                                <034>
610 NEXT A                                    <039>
619 REM                                       <252>
620 REM LAUFSCHRIFT SETZEN                    <017>
621 REM                                       <254>
625 FOR LP=1 TO LEN(LA$)-25                   <238>
630 LZ=LZ-1:IF LZ>0 THEN POKE P4+22,LZ OR 8
   :FOR A=0 TO 9:NEXT A:GOTO 630              <060>
640 PRINT TAB(6);:WAIT 53265,128:WAIT 53266,64
   :POKE 53206,15:PRINT LF$:PRINT"[UP2]"      <125>
660 LZ=7:LF$=MID$(LA$,LP,25)                  <238>
670 NEXT LP                                   <190>
680 GET A$:IF A$=""THEN 500                   <012>
690 SYS 52970:REM PROVIC-64 DESAKTIVIEREN     <090>
999 REM                                       <121>
1000 REM SPRITE-DATEN                         <234>
1001 REM                                      <123>
1002 DATA 0,0,0,0,126,0,1,255,128,7,255,224,15,
   255,240,15,253,240,31,255,248              <008>
1003 DATA 31,255,248,63,255,252,63,255,252,63,
   243,252,63,252,0,63,255,252,63             <055>
1004 DATA 255,252,31,255,248,31,255,248,15,255,
   240,15,255,240,7,255,224,1,255             <099>
1005 DATA 128,0,126,0,0,0,0,0,0,126,0,1,255,128,
   7,255,224,15,255,240,15,251                <197>
1006 DATA 240,31,255,248,31,255,248,63,255,240,
   63,255,0,63,240,0,63,252,0,63              <036>
1007 DATA 255,0,63,255,224,31,255,248,31,255,
   248,15,255,240,15,255,240,7,255            <053>
1008 DATA 224,1,255,128,0,126,0               <119>
```

2000 Hamburg

SOFTWARE · HARDWARE
commodore
BDB BÜRO KG
EIFFESTRASSE 398
2000 HAMBURG 26
TEL. (040) 25 1605-0

G.P.O. Micro-Computer-Laden
Vertragshändler für commodore
Softwarepartner von: TA
Wir führen: alphatronic PC, P2, P3, P4
commodore VC 20 u. VC 64
Atari 400, 800, 600 XL + 800 XL
Philips G 7000 —
u. viel, viel Zubehör
G.P.O. GmbH Micro Computer Laden
Schulweg 25a - 2000 Hamburg 19
Tel.: 040/406610
Lohbrügger Landstr. 80-82 -
2050 Hamburg 80, Tel. 040/7389277
Elektronik Zubehör

2390 Flensburg



D. Lassen · U. Fahrensohn
elektronic computer laden ohg
Norderstr. 94-96 Flensburg 28181

2400 Lübeck

Ingenieurbüro
CHRISTINE MOEBIUS
Computer & Software
Würen 38, 2350 Neumünster
Tel. 04321/71623
Resselweg 8, 2400 Lübeck
Tel. 0451/594206

3000 Hannover



4620 Castrop-Rauxel

COMMODORE REPARATUREN
AVC
extrem schnell u. preiswert
Bahnhofstraße 84-86
462 Castrop-Rauxel
Tel.: 02305/37 47 o. 38 48
AUDIO-VIDEO-COMPUTER-SERVICE

5000 Köln



5060 Bergisch Gladbach

commodore
Vertragshändler
COMPUTER CENTER
Norbert Stellberg
Buchholzstraße 1,
5060 Bergisch Gladbach
Telefon 02202/35053

5063 Overath

commodore
Vertragshändler
COMPUTER CENTER
Norbert Stellberg
Blindenaaf 36
5063 Overath
Telefon 02206/6644

5200 Siegburg

commodore
Vertragshändler
COMPUTER CENTER
Norbert Stellberg
Luisenstraße 26,
5200 Siegburg
Telefon 02241/66854

5270 Gummersbach

commodore
Vertragshändler
COMPUTER CENTER
Norbert Stellberg
Marktstraße 9,
5270 Gummersbach
Telefon 02261/22855

6000 Frankfurt



6457 Maintal-Dörnigheim



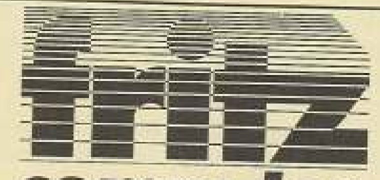
6740 Landau



6800 Mannheim

Ihr
commodore -Partner
für Rhein/Neckar + Rhein-Pfalz
GAUCH + STURM
Computersysteme + Textsysteme
Casterfeldstr. 74-76
6800 Mannheim 24
Tel. (0621) 852045, Teletex 6211912

7000 Stuttgart



computer
- MDT-Micro-Computer
- Bildschirm-Textsysteme
- Speicher-Schreibmaschinen
- Software-Rund-Um-Service

Fritz Computer GmbH
Schulze-Delitzsch-Str. 22, D-7000 Stuttgart 80,
Tel. (0711) 7800-238

7038 Holzgerlingen

CBM 64 — Joachim Zaich — ACORN
Das be ... nende
Softwarehaus
Tübinger Str. 3 * 7038 Holzgerlingen
☎ 07031 44002 * Individualprogrammierung
Hard- + Software * Eigene Spielesoftware
Fachbücher * Zubehör * 3M Disketten
Verkauf Geöffnet: Samstag 9-14 Uhr
Büro Geöffnet: nach Vereinbahrung

7150 Backnang



7700 Singen



Schellhammer
7700 Singen, Freibühlstr. 21-23,
Tel. 07731/82020, Tx.: 0793716

7730 VS-Villingen

MAIER
Datensysteme GmbH
commodore
Vertragshändler
- Beratung · Verkauf
- Service · Software
- Interface-Entwicklung

Tel. 07721/70322
Gnädlingstraße 5
7730 Villingen-Weilersbach

8000 München

System-Händler
commodore COMPUTER
HDS PRÜFTECHNIK
HDS-PRÜFTECHNIK GmbH
Maria-Eich-Straße 1,
8000 München 60,
Telefon (089) 837021/22

schulz computer
Beratung - Verkauf - Kundendienst
Commodore - IBM - Epson
Hewlett-Packard - EDV-Zubehör
Microcomputer Bauteile -
Literatur

Schulz computer
Schillerstr. 22, 8000 München 2
Telefon 089/597330

8200 Rosenheim

IHR PARTNER FÜR ...
- Commodore-Computer
- Micro- und Personalcomputer
- Anwendersoftware + Spiele
- Computer-Zubehör + Literatur
- Fachmännische Beratung

Commodore MARABU ELECTRONICS
H. HÖBELER-KOHLHEPP
Hochgernstr. 2 (Haus am Brückenberg)
8200 ROSENHEIM/Obb.
Telefon 08031/45784*

SCHWEIZ

Luzern

DCT Computer Shop Luzern
Auch für C-64 die
Nr. 1 in der Zentralschweiz
Seeburgstrasse 18, 6002 Luzern
☎ 041-31 45 45

Das intelligente Algebraprogramm
für den COMMODORE 64
Macht die Mathematik zum Kinderspiel

Heureka-Software
Vom Lehrer für Schüler



sagt Euren Eltern, daß es ALI gibt!
ALI gibt's auf Diskette mit Anleitung
und mehr als 100 abrufbaren Beispielen.

ALI

Unterstützt den Mathematikunterricht
am Gymnasium
von der Unterstufe bis zum Abitur

99.– DM unverbindliche Preisempfehlung — jetzt auch bei Quelle

ALI löst Aufgaben wie sie im Mathematikbuch stehen - nachvollziehbar, Schritt für Schritt:

Vom Mathematikbuch einfach in den Rechner
und ALI zeigt Dir wie's geht

ALI - rechnet verschachtelte Klammern aus
ALI - vereinfacht komplizierte Terme
ALI - zerlegt in Faktoren
ALI - löst Gleichungen und Ungleichungen

© Dipl.-Phys. Peter Ostermann,
Wastl-Witt-Str. 46, 8000 München 21
Telefon 089/706383

Der Autor ist seit mehr als 10 Jahren Mathematiklehrer am Gymnasium

Hypra-Load mal vier

Selten hat ein Programm so viele Leserreaktionen hervorgerufen wie Hypra-Load. Deswegen führen wir Ihnen heute vor, was man mit Hypra-Load noch so alles machen kann.

Hätten wir doch bloß nicht unsere Telefonnummern angegeben. Das war unser beherrschender Gedanke in den ersten Tagen nach Erscheinen der Ausgabe 10/84 des 64'er-Magazins. Doch nach und nach begannen wir am Telefonieren Gefallen zu finden, wohl auch deswegen, weil fast alle Anrufer so nett und höflich waren, und uns einige gute Tips und konstruktive Kritik gaben. Uns erreichten sogar Telefonate aus der Schweiz, Österreich, Holland und Dänemark.

Viele Anrufer hatten ein gemeinsames Problem: Nach RUN meldete sich der Basic-Lader immer mit FEHLER IN DATENBLOCK 100-109, obwohl dort garantiert kein Fehler vorlag. Manche Leser haben diesen DATA-Block mehrere Male eingetippt, und sogar von Hand aufaddiert und verglichen, konnten aber keinen Fehler finden. Hatte vielleicht unser Druckfehlerteufelchen zugeschlagen? Nein, das hatte genug Respekt vor der DATA-Wüste, um sie zu verschonen. Des Rätsels Lösung: Uns unterlief ein Denkfehler im Prüfsummenprogramm. Denn das arbeitet nur dann korrekt, wenn alle Zeilen und alle Kommata eingegeben werden, sonst meldet die total verwirrte Prüfsummenroutine einen Fehler, wo gar keiner ist. Entgegen unserer Aussage, daß Hypra-Load mit dem SX 64 nicht funktioniert, hatten einige SX 64-Besitzer keine Probleme. Es scheint drei oder vier verschiedene SX 64-Versionen zu geben, von denen sich mindestens eine nicht mit Hypra-Load verträgt. Wir wollen die genauen Gründe noch erforschen, und bitten alle SX 64-Besitzer um ihre Hilfe. Schreiben sie uns ihre Erfahrungen mit ihrem SX 64. Welche Programme laufen, welche nicht? Insbesondere würde uns dann die Seriennummer ihres SX 64 interessieren.

Unsere Bemühungen, Hypra-Load für den VC 20 umzuschreiben, scheiterten bisher an dessen Busstruktur.

Ein Fehler schlich sich in der Beschreibung zu Hypra-Load ein. Um es nach einem RESET mit einem Taster wieder einzuschalten, muß als erstes POKE 40960, PEEK (40960) eingegeben werden. Dann darf mit POKE 1,53 eingeschaltet werden.

So, nach dieser langen Vorrede aber endlich zu den versprochenen Änderungen an Hypra-Load.

Hypra-Boot

Manche Leser ärgerten sich darüber, daß, wolle man ein bestimmtes, langes Programm schnell laden, zwei LOAD- und zwei RUN-Befehle notwendig sind. Deswegen haben wir Hypra-Load zu Hypra-Boot umgeschrieben. Wie funktioniert Hypra-Boot? Nach dem Laden und Starten von Hypra-Boot wird automatisch ein weiteres Programm von Diskette nachgeladen und gestartet. Dieses Programm muß, wenn es kein Basic, sondern ein Maschinenprogramm ist, allerdings am Basic-Start mit zumindest einem SYS-Befehl vertreten sein, da Hypra-Boot den RUN-Befehl simuliert.

Und so geben sie Hypro-Boot ein: Laden Sie den Basic-Lader, das heißt, das in Ausgabe 10/84 abgedruckte Programm. Geben Sie nun die in Listing 1 abgedruckten Änderungen ein.

Damit Sie beliebige Programme nachladen können, müssen Sie natürlich noch den Filenamen des nachzuladenden Programms angeben. Dies geschieht wie folgt in der Zeile 276. Geben Sie als DATA-Werte die ASCII-Codes der ersten sieben Buchstaben des Filenamens und als achten Wert 042 (ASCII-Code für *) ein. Kürzere Filenamen sind erlaubt, dann muß der Stern aber auch früher stehen, soll kein FILE NOT FOUND ERROR gemeldet werden.

Da die DATA-Zeilen schon mit Prüfsummen für unseren Checksummer versehen sind, nimmt das Programm selbst keinen Prüfsummencheck vor!

Es wird auch nicht mit RUN, sondern mit GOTO 520 gestartet. Danach speichern Sie Hypra-Boot unter dem gewünschten Namen.

Hypra-Boot kann nicht nur Einzelprogramme, sondern zum Beispiel auch ein Menü-Programm oder ähnliches laden.

Ganz Verwegene können Hypra-Boot noch mit einem Autostart versehen. Allerdings ist Vorsicht geboten, sollen Autostart-Programme nachgeladen werden. Wenn diese den Stack überschreiben, kann es Konflikte mit Hypra-Boot geben. Am besten, Sie probieren es einfach aus. Hypra-Boot lädt immer wie LOAD"name",8,1!

Es kann ab und zu passieren, das Hypra-Boot beim Nachladen »VERIFYING« anzeigt. Dies hat nichts zu bedeuten und tritt nur auf, wenn Hypra-Boot selber ein Autostart-Programm ist. Es empfiehlt sich, Hypra-Boot so zu nennen, wie das nachzuladende Programm. Dieses erhält dann einen Namen wie "!01" oder ähnliches. Dann können mehrere Hypra-Boot-Programme auf einer Diskette stehen.

Sollten Sie den Basic-Lader nicht besitzen, so wird es etwas kompliziert. Sie müssen, nach dem Laden von Hypra-Load, die entsprechenden DATA-Werte von Hand an die Speicherstellen von 3385 bis 3464 POKEn. Sie tippen also ein:

```
POKE 3385, 245
POKE 3386, 169
...
POKE 3464, 000
```

Vergessen Sie dabei nicht, sich vorher die DATA-Werte für den Filenamen zu notieren!

Hypra-Track 18

Wer sich weder ein Modul brennen, noch das Betriebssystem austauschen möchte, wäre froh, wenn er Hypra-Load auf jeder Diskette abspeichern könnte, um sich Diskettenwechsel zu ersparen. Aber da gibt es doch immer einige Disketten, die

sich strikt weigern, Hypra-Load noch aufzunehmen, da weniger als 6 Blöcke frei sind. Was soll man tun? Weiterhin Disketten wechseln, den Inhalt auf zwei Disketten verteilen, oder auf Hypra-Load verzichten?

Es gibt noch eine letzte Rettung: Hypra-Load wird ins Directory-Track geschrieben. Normalerweise passen in ein Directory 144 Einträge. Wenn Hypra-Load am Ende des Directory-Tracks steht, ist dieses auf 88 Einträge begrenzt. Die normale Funktion des Directory wird dadurch nicht beeinträchtigt! Das Programm überprüft natürlich, ob weniger als 88 Einträge vorhanden sind, damit nichts aus Versehen gelöscht wird.

Das als Listing 2 abgedruckte Programm schreibt Hypra-Load in folgende Sektoren der Spur 18: 3,6,9,12,15,18.

Der Filename, den Hypra-Load tatsächlich erhalten soll, ist frei wählbar. An diesen wird »",8:« angehängt. Somit ist es möglich, bei gelistetem Directory nur mit dem Cursor auf diesen Fileeintrag zu fahren, LOAD zu schreiben und RETURN zu drücken. Nach dem Laden startet man es, wie gewöhnlich, mit RUN.

Und so geben sie Listing 2 ein:

1. Laden Sie den Basic-Lader von Hypra-Load, abgedruckt in Ausgabe 10/84
2. Löschen Sie alle Zeilen außer die Zeilen von 100 bis 279
3. Geben Sie nun Listing 2 zusätzlich ein.
4. Speichern Sie das Ganze auf Diskette.

Wir empfehlen, unbedingt einen Testlauf mit einer leeren formatierten Diskette zu machen, da sich Tippfehler eingeschlichen haben könnten. Es wird kein Prüfsummenvergleich vorgenommen! Nach erfolgreichem Testlauf kann Hypra-Load auf praktisch alle Disketten überspielt werden.

Hypra-ROM

Manchmal ärgerten sogar wir uns darüber, daß Hypra-Load dann und wann neu geladen werden muß, sei es, weil der Computer eingeschaltet wurde, sei es, weil irgendein Programm mal wieder Hypra-Load oder das Basic-RAM überschrieben hatte. Hypra-Load resident machen, hieß also die Devise. Und wenn man nicht gerade das Betriebssystem auswechseln möchte, bietet es sich geradezu an, Hypra-Load in ein Modul zu brennen.

Hierzu sind weit weniger Änderungen nötig, als erwartet. Ganze vier DATA-Werte müssen modifiziert werden, soll Hypra-Load nicht ab $0800 sondern ab $8000 im Speicher stehen.

Es sind dies folgende Werte:

Zeile	Position	alter Wert	neuer Wert
256	5	010	130
260	4	008	128
264	4	013	133
266	1	012	132

In den entsprechenden Bereich schreiben Sie Hypra-Load wie folgt:

FOR I=32769 TO 34207 : READ A : POKE I,A : NEXT I

Bitte führen sie keinen RUN aus, da ja nun die Checksummen nicht mehr stimmen. Gestartet wird das Ganze nun mit SYS 33958. Hypra-Load meldet sich wie gewohnt. Dieses Programm kann auch in ein Modul gebrannt werden. Auf einen Autostart im Modul wurde verzichtet, da es ja einige Programme gibt, die sich mit Hypra-Load nicht vertragen (zum Beispiel Simons Basic).

Hier noch die POKEs für alle, die nur das Maschinenprogramm haben:

POKE 3301, 130: POKE 3332, 128
POKE 3356, 133: POKE 3343, 132

Danach muß der Speicherbereich von $0800 bis $0E00 nach $8000 verschoben werden, am besten mit einem Monitor oder einer FOR-NEXT-Schleife.

Dieser Bereich kann nun auch in ein EPROM gebrannt werden.

Wer Hypra-Load in andere Bereiche verschieben möchte, der sollte sich mit einem Disassembler das Umfeld der vier angegeben Adressen ansehen, und die Angleichungen selber vornehmen. Zu beachten ist nur, daß all diese Versionen immer noch das Betriebssystem ins RAM kopieren und dort verändern. Auf diese Art und Weise ist es nicht möglich, das Betriebssystem im ROM zu ändern!

Schnelles Laden ohne Laden

Das Arbeiten mit Hypra-Load und dem Diskettenlaufwerk 1541 ist für viele Commodore-Besitzer schon zur unverzichtbaren Gewohnheit geworden. Leider muß das Programm zuvor immer noch von Diskette geladen werden. Fest im Betriebssystem einprogrammiert entfällt dieser Nachteil.

Jedes Programm, daß die Datenübertragungsgeschwindigkeit vom Diskettenlaufwerk zum Computer beschleunigt, büßt einen Teil seines Geschwindigkeitsvorteils ein, weil es erst geladen und gestartet werden muß. Wesentlich angenehmer ist es, wenn der Computer schon nach dem Einschalten die gewünschte Laderoutine verwenden kann. Um zu diesem Ziel zu gelangen, sind zwei Wege denkbar. Zum einen kann man sich, ähnlich einem Spielmodul, eine Steckplatine mit EPROMs bauen und diese am Expansion-Port anschließen. Zum anderen besteht aber die Möglichkeit, den Computer intern zu verändern, indem ein neues Betriebssystem eingebaut wird. Die Vorteile der zweiten Methode sind beachtlich.

Dadurch, daß die Änderung des Betriebssystems nicht mehr im RAM-Bereich von $E000-$FFFF liegt, sondern im gleichen, darüberliegenden ROM-Bereich, steigt die Kompatibilität mit jeder Art von Programmen enorm. Die Erklärung dafür ist einfach. Viele Programme verwenden den $E000-Bereich, indem sie ihn kopieren und für ihre eigenen Zwecke modifizieren. Jedes dort befindliche Programm wird damit selbstverständlich überschrieben. Dazu gehören aber leider immer Betriebssystemänderungen wie beispielsweise das Hypra-Load. Steht die Veränderung aber im darüberliegenden ROM, so entstehen diese Probleme kaum noch, denn der beschriebene RAM-Bereich steht, wie bei jedem Original-Betriebssystem, frei zur Verfügung.

Natürlich ist zum Einbau des neuen Betriebssystems ein Eingriff in den Computer notwendig. Das alte Kernal-ROM muß gegen das neue ausgetauscht werden. Wie das gemacht wird, haben wir in der letzten Ausgabe ausführlich beschrieben. Die wichtigsten Arbeitsschritte werden aber trotzdem nocheinmal erklärt. Zunächst ist es notwendig, das abgebildete Basic-Programm einzugeben und zu starten. Wenn es fehlerfrei funktioniert, kann es abgespeichert werden. Als nächstes wird ein EPROM-Programmiergerät benötigt, das in der Lage ist, EPROMs vom Typ 2764 zu brennen. Dieses Programmiergerät wird nun angeschlossen. Der nächste Arbeitsgang besteht

aus der eigentlichen Veränderung des Betriebssystems. Alle notwendigen Programmierungen werden vom abgedruckten Programm selbständig durchgeführt. Dazu wird zunächst der Bereich $E000-$FFFF nach $6000-$7FFF kopiert. In Overlaytechnik werden anschließend sowohl das Hypra-Load als auch eine Funktionstastenbelegung programmiert. Wer auf die Funktionstastenbelegung verzichten möchte, braucht im Listing übrigens nur die Zeilen 8000 bis 9000 wegzulassen. Nachdem das Programm durchgelaufen ist, befindet sich im Bereich $6000 bis $7FFF ein komplettes, neues Betriebssystem, das lediglich noch in ein EPROM gebrannt werden muß. Selbstverständlich darf die Treibersoftware für das EPROM-Programmiergerät nicht in diesem Bereich liegen.

Das fertig gebrannte EPROM wird einfach im Steckplatz U4 auf der Computerplatine mit einem Adapter eingesteckt. Der Adapter ist leider wegen der unterschiedlichen Pinbelegung der Commodore-ROMs und des EPROMs unabdingbar. Auch wenn die Herstellung eines solchen Zwischenstücks etwas Geschick erfordert, stellt sie doch kein unüberbrückbares Problem dar.

Wenn der Computer nun nach dem Einschalten seine Bereitschaftsmeldung zeigt, steht dem schnellen Laden nichts mehr im Weg. Andernfalls sollten Sie das EPROM löschen und den Vorgang wiederholen. Die Belegung der Funktionstasten entspricht der in der letzten Ausgabe veröffentlichten, hinzugekommen ist aber die Belegung der RUN-Taste. Durch einmaliges Betätigen sind wieder die langsamen Laderoutinen aktiviert. Ein zweiter Druck auf die gleichen Tasten schaltet wieder auf Hypra-Load. Leider muß gelegentlich, besonders bei Autostartprogrammen, von dieser Funktion Gebrauch gemacht werden. Auch sollte zum Laden von Programmen nur ein Peripheriegerät eingeschaltet sein, da sonst eine Fehlermeldung ausgegeben wird. Die Belegung der Funktionstasten kann bei Bedarf mit Poke 2,1 ausgeschaltet und mit Poke 2,0 wieder angeschaltet werden.

Abgesehen von diesen Nachteilen und den nicht mehr möglichen Betrieb eines Datenrecorders, läßt sich mit dem neuen Betriebssystem sehr gut arbeiten. In einer der nächsten Ausgaben werden wir dem Commodore 64 und dem MPS 802 zusätzlich noch einen deutschen Zeichensatz verleihen. Schnelles Laden und eine Funktionstastenbelegung — der Commodore macht sich.

(Günther Reimuth/Uwe Schönewolf/Boris Schneider/ Erich Schöberl/Arnd Wängler/gk)

Listing 1. Hypra-Boot lädt und startet ein ausgewähltes Programm automatisch.

```
267 DATA 245,169,053,133,001,169,076,162   <148>
268 DATA 115,160,228,141,045,228,142,046   <138>
269 DATA 228,140,047,228,076,248,252,000   <145>
270 DATA 000,000,000,000,000,000,169,016   <093>
271 DATA 162,157,160,228,032,249,253,162   <151>
272 DATA 008,160,001,032,000,254,032,213   <115>
273 DATA 255,165,174,133,045,165,175,133   <157>
274 DATA 046,032,099,166,032,142,166,076   <158>
275 DATA 174,167,000,000,000,000,000,000   <101>
276 DATA 000,000,000,000,000,000,000,000   <076>
277 DATA 000,000,000,000,000,000,000,000   <077>
```

Listing 2. Hypra-Load belegt keinen sichtbaren Speicherplatz mehr auf Diskette.

```
1000 CLR                                      <135>
1010 REM                                      <132>
1020 REM ******************************** <206>
1030 REM ***                          ***     <149>
1040 REM ***        HYPRA-LOAD        ***     <112>
1050 REM *** MIT 0 BLOECKEN AUF DISK  ***     <013>
1060 REM ***                          ***     <179>
1070 REM ***            BY            ***     <088>
1080 REM ***     UWE SCHOENEWOLF      ***     <245>
1090 REM ***     GUENTHER REIMUTH     ***     <081>
1100 REM ***                          ***     <219>
1110 REM ******************************** <041>
1120 REM                                      <243>
1130 POKE 53280,0:POKE 53281,6                <153>
1140 NA$=CHR$(130)+CHR$(18)+CHR$(3)           <169>
1150 NB$=",8:"                                <202>
1160 PRINT"{CLR,WHITE,DOWN,2SPACE}HYPRA LOAD
    MIT 0 BLOCKS AUF DISKETTE"                <199>
1170 PRINT"{DOWN,10SPACE,GREY 3}VON
   : UWE SCHOENEWOLF"                         <122>
1180 PRINT"{10SPACE,GREY 3,5SPACE}
   GUENTHER REIMUTH{YELLOW}"                  <054>
1190 A$="":INPUT"{3DOWN,SPACE}HYPRA LOAD AUF
    DISKETTE SCHREIBEN{2RIGHT,2SPACE}J{3LEFT}";
   A$                                         <166>
1200 IF A$="J"THEN PRINT"{2DOWN}":GOTO 1220
                        <018>
1210 SYS 64738                                <104>
1220 OPEN 1,0                                 <244>
1230 N$="":PRINT"{2UP,DOWN,2RIGHT}FILE NAME
    MAX. 1-12 >HYPRA LOAD{10LEFT}";:INPUT#1,N$
                        <160>
1240 LA=LEN(N$)                               <161>
1250 CLOSE 1                                  <183>
1260 IF LA>12 OR LA<1 THEN 1220               <169>
1270 OPEN 1,8,15                              <192>
1280 OPEN 2,8,2,N$                            <054>
1290 INPUT#1,A                                <049>
1300 CLOSE 2                                  <235>
1310 CLOSE 1                                  <244>
1320 IF A=0 THEN PRINT:PRINT"{2DOWN,SPACE,RVSON,
   SPACE}FILE SCHON VORHANDEN{SPACE,RVOFF}"
   :GOTO 1670                                 <201>
1330 NA$=NA$+N$+CHR$(160)+NB$                 <034>
1340 LA=LEN(NA$)                              <071>
1350 FOR I=19-LA TO 1 STEP-1                  <195>
1360 NA$=NA$+CHR$(160)                        <198>
1370 NEXT                                     <225>
1380 FOR I=1 TO 12                            <029>
1390 NA$=NA$+CHR$(0):NEXT                     <057>
1400 REM ***********************              <126>
1410 REM * TEST AUF FREIRAUM *                <225>
1420 REM * FUER HYPRA-LOAD   *                <119>
1430 REM ***********************              <156>
1440 T=18:S=1:Y=1                             <250>
1450 OPEN 1,8,15                              <117>
1460 OPEN 2,8,2,"#"                           <223>
1470 PRINT#1,"U1 2 0";T;S                     <001>
1480 PRINT#1,"B-P";2;0                        <157>
1490 GET#2,T1$:IF T1$=""THEN T1$=CHR$(0)      <080>
1500 GET#2,S1$:IF S1$=""THEN S1$=CHR$(0)      <087>
1510 T1=ASC(T1$):S1=ASC(S1$)                  <017>
1520 IF T1<>0 THEN T=T1:S=S1:Y=Y+1:GOTO 1470
                        <150>
1530 IF Y>12 THEN PRINT:PRINT"{2DOWN,SPACE,
   RVSON,SPACE}KEIN PLATZ MEHR AUF DISKETE
   {SPACE,RVOFF}":GOTO 1570                   <103>
1540 CLOSE 2                                  <220>
1550 CLOSE 1                                  <229>
1560 GOTO 1630                                <113>
1570 CLOSE 2                                  <250>
1580 CLOSE 1                                  <003>
1590 GOTO 1670                                <147>
1600 REM ******************************       <151>
```

```
1610 REM * H-LOAD AUF DISK SCHREIBEN *       <098>
1620 REM *****************************       <171>
1630 GOSUB 1730                              <188>
1640 GOSUB 2010                              <190>
1650 PRINT                                   <017>
1660 PRINT"{DOWN,SPACE}FERTIG"               <049>
1670 PRINT"{DOWN,SPACE}WEITER MIT [ SPACE ]"
                     <088>
1680 GET A$:IF A$=""THEN 1680                <050>
1690 GOTO 1130                               <238>
1700 REM *****************                   <045>
1710 REM * NAME SCHREIBEN *                  <075>
1720 REM *****************                   <065>
1730 T=18:S=1:Y=1                            <029>
1740 OPEN 1,8,15                             <152>
1750 OPEN 2,8,2,"#"                          <002>
1760 PRINT#1,"U1 2 0";T;S                    <036>
1770 PRINT#1,"B-P";2;0                       <192>
1780 GET#2,T1$:IF T1$=""THEN T1$=CHR$(0)     <115>
1790 GET#2,S1$:IF S1$=""THEN S1$=CHR$(0)     <122>
1800 T1=ASC(T1$)                             <006>
1810 S1=ASC(S1$)                             <014>
1820 FOR Q=0 TO 7                            <178>
1830 PRINT#1,"B-P";2;Q*32+2                  <011>
1840 GET#2,X$                                <213>
1850 IF X$=""THEN X$=CHR$(0)                 <091>
1860 IF ASC(X$)=0 THEN 1960                  <194>
1870 NEXT                                    <215>
1880 IF T1<>0 THEN T=T1:S=S1:Y=Y+1:GOTO 1760
                     <002>
1890 IF Y>12 THEN PRINT:PRINT"{2DOWN,SPACE,
   RVSON}ZUVIELE FILES AUF DER DISKETTE{RVOFF}"
   :GOTO 1990                                <191>
1900 IF Q<8 THEN 1960                        <177>
1910 PRINT#1,"B-P";2;0                       <077>
1920 PRINT#2,CHR$(T)+CHR$(S+3);              <022>
1930 PRINT#1,"U2 2 0";T;S                    <208>
1940 S=S+3                                   <208>
1950 PRINT#1,"U1 2 0";T;S                    <227>
1960 PRINT#1,"B-P";2;Q*32+2                  <141>
1970 PRINT#2,NA$;                            <157>
1980 PRINT#1,"U2 2 0";T;S                    <002>
1990 CLOSE 2:CLOSE 1:RETURN                  <114>
2000 REM *********************               <216>
2010 REM * DATAS ABSPEICHERN *               <085>
2020 REM *********************               <236>
2030 RESTORE                                 <129>
2040 OPEN 1,8,15                             <197>
2050 OPEN 2,8,2,"#"                          <048>
2060 FOR S=3 TO 15 STEP 3                    <179>
2070 PRINT#1,"B-P 2 0"                       <120>
2080 A$="":A$=A$+CHR$(18)+CHR$(S+3)          <054>
2090 PRINT#2,A$                              <141>
2100 PRINT#1,"B-P 2 2"                       <152>
2110 A$=""                                   <161>
2120 EN=253                                  <047>
2130 IF S=3 THEN PRINT#2,CHR$(1);CHR$(8);:EN=251
                     <224>
2140 FOR I=0 TO EN                           <071>
2150 READ X                                  <077>
2160 A$=A$+CHR$(X)                           <014>
2170 NEXT                                    <004>
2180 PRINT#2,A$;                             <034>
2190 PRINT#1,"U2 2 0 18";S                   <175>
2200 NEXT                                    <034>
2210 PRINT#1,"B-P 2 0"                       <004>
2220 A$="":A$=A$+CHR$(0)+CHR$(170)           <017>
2230 FOR I=1 TO 170                          <167>
2240 READ X                                  <167>
2250 A$=A$+CHR$(X)                           <104>
2260 NEXT                                    <094>
2270 PRINT#2,A$;                             <124>
2280 PRINT#1,"U2 2 0 18";S                   <009>
2290 CLOSE 2:CLOSE 1:RETURN                  <159>
```

Listing 3. Hypra-Load fest in ein neues Betriebssystem eingebunden.

```
600 POKE 56,96:POKE 55,0:CLR                 <100>
890 PRINT"KOPIEREN DES ROM INS RAM":PRINT    <030>
900 FOR I=6*4049 TO 8*4096-1
   :POKE I,PEEK(I+32768):NEXT I              <072>
999 REM DATA ZEILEN EINLESEN                 <173>
1000 PRINT"{CLR}":PRINT:PRINT TAB(10);"LESEN
   DER DATA-ZEILEN":PRINT:PRINT              <179>
1010 T=0:OF=32768                            <182>
1020 T=T+1:READ A:IF A=0 THEN 1100           <135>
1030 READ B:REM ANZAHL DER BYTES             <188>
1032 READ P1:REM PRUEFSUMME                  <230>
1033 P2=0                                    <113>
1035 PRINT"BLOCK ";T;"{2SPACE}";             <160>
1040 FOR I=A-OF TO A-OF-1+B                  <254>
1050 READ D:POKE I,D                         <115>
1051 P2=P2+D:IF P2>65535 THEN P2=P2-65536    <212>
1053 NEXT I                                  <236>
1055 IF P2<>P1 THEN 1070                     <132>
1057 PRINT"{3SPACE}OK"                       <156>
1060 GOTO 1020                               <116>
1070 PRINT"PRUEFSUMME FALSCH: ";P2           <192>
1080 GET A$:IF A$=""THEN 1080                <210>
1090 GOTO 1020                               <146>
1100 PRINT"FERTIG":END                       <168>
7990 REM AB HIER DATAS                       <252>
7995 REM****************************************
                     <121>
7996 REM **ZEILEN 8000 - 9000 KOENNEN FUER BETR
   IEBSSYSTEM OHNE FUKTIONSTASTEN**          <248>
7997 REM **WEGGELASSEN WERDEN**              <141>
8000 DATA 64226,126,14014                    <209>
8001 DATA 232,134,198,201,133,144,4,201,141,144,
   3,76,66,235,157,119,2,72,152              <150>
8002 DATA 72,160,0,196,2,208,13,185,41,251,221,
   119,2,240,11,200,192,176,208              <087>
8003 DATA 243,104,168,104,76,66,235,200,185,41,
   251,201,133,144,4,201,141,144             <140>
8004 DATA 238,236,137,2,176,233,157,119,2,232,
   134,198,76,15,251,133,76,207              <080>
8005 DATA 34,36,34,44,56,13,137,76,79,65,68,134,
   76,73,83,84,13,138,83,65,86               <160>
8006 DATA 69,135,82,85,78,13,139,83,89,83,136,
   63,70,82,69,40,48,41,13,140,83            <202>
8007 DATA 89,83,54,52,55,51,56,13,133,136    <241>
8009 REM ***************************************
   **                                        <135>
8010 DATA 60223,3,552                        <016>
8011 DATA 76,226,250                         <227>
9000 REM ***************************************
   **                                        <106>
9010 DATA 58552,33,4800:REM $E4B8-$E4D8      <145>
9020 DATA 133,147,169,0,133,144,165,186,208,4,
   169,8,133,186,201,3,144,248,164           <220>
9030 DATA 183,208,3,76,16,247,166,185,76,145,
   248,234,234,234                           <173>
9040 REM ***************************************
   **                                        <146>
9050 DATA 60647,9,530:REM $ECE7-$ECEF        <133>
9060 DATA 83,89,83,54,52,53,55,48,13         <035>
9070 REM ***************************************
   **                                        <176>
9080 DATA 62637,23,3533:REM $F4AD-$F4C3      <222>
9090 DATA 201,4,176,4,169,8,133,186,164,183,208,
   3,76,16,247,166,185,234,234               <201>
9100 DATA 234,234,234,234                    <026>
9110 REM ***************************************
   **                                        <216>
9120 DATA 63276,212,29783:REM$F72C-$F7FF     <129>
9130 DATA 169,226,162,248,133,167,134,168,169,0,
   162,3,133,169,134,170,165,186             <087>
9140 DATA 32,12,237,169,111,32,185,237,169,77,
   32,221,237,169,45,32,221,237              <191>
9150 DATA 169,87,32,221,237,160,0,165,169,32,
   221,237,165,170,32,221,237,169            <248>
```

```
9160 DATA 30,32,221,237,177,167,32,221,237,200,
   192,30,144,246,32,254,237,24             <238>
9170 DATA 165,167,105,30,133,167,144,3,230,168,
   24,165,169,166,170,105,30,133            <051>
9180 DATA 169,144,2,230,170,224,5,144,173,201,0,
   144,169,165,186,32,12,237,169            <110>
9190 DATA 111,32,185,237,169,77,32,221,237,169,
   45,32,221,237,169,69,32,221              <246>
9200 DATA 237,169,139,32,221,237,169,4,32,221,
   237,169,239,45,17,208,32,90,252          <151>
9210 DATA 234,234,76,21,252,169,11,141,0,221,44,
   0,221,16,251,169,3,141,0,221             <054>
9220 DATA 162,5,202,234,208,252,162,4,173,0,221,
   42,42,102,176,106,102,176,234            <124>
9230 DATA 202,208,242,165,176,73,255,96,32,195,
   247,201,255,240,248,160,0,169            <129>
9240 DATA 11,141,0,221,44,0,221,16,251      <008>
9250 REM **************************************
   **                                       <101>
9260 DATA 63488,256,34399:REM $F800-F8FF    <227>
9270 DATA 169,3,141,0,221,162,7,202,208,253,173,
   0,221,42,42,102,176,106,102              <066>
9280 DATA 176,234,234,173,0,221,42,42,102,176,
   106,102,176,234,234,173,0,221            <090>
9290 DATA 42,42,102,176,106,102,176,234,234,173,
   0,221,42,42,102,176,106,102              <091>
9300 DATA 176,165,176,73,255,153,0,2,200,208,
   180,96,120,169,1,133,167,160,255         <236>
9310 DATA 32,237,247,192,255,240,64,162,2,165,
   167,240,2,162,4,173,0,2,208,7            <135>
9320 DATA 238,1,2,173,1,2,44,169,0,133,168,189,
   0,2,145,174,230,174,208,2,230            <183>
9330 DATA 175,232,228,168,208,240,162,0,134,167,
   173,0,2,208,198,169,16,13,17             <219>
9340 DATA 208,234,141,17,208,169,64,133,144,24,
   96,32,175,245,169,96,133,185             <202>
9350 DATA 32,213,243,165,186,32,9,237,165,185,
   32,199,237,32,19,238,133,174             <159>
9360 DATA 165,144,74,74,144,3,76,48,245,32,19,
   238,133,175,138,208,20,165,195           <013>
9370 DATA 133,174,165,196,133,175,32,210,245,
   169,253,37,144,133,144,76,160            <168>
9380 DATA 251,165,175,201,4,176,238,76,240,244,
   255,255,255,255,255,255,255              <191>
9390 DATA 255,234,96,165,0,41,6,201,2,240,3,76,
   158,253,234,169,5,133,9,162              <174>
9400 DATA 90,134,75,162,0,169,82,133,36,32,86,
   245,80                                   <133>
9410 REM **************************************
   **                                       <005>
9420 DATA 63744,256,27098:REM$F900-$F9FF    <162>
9430 DATA 254,184,173,1,28,197,36,240,9,198,75,
   208,239,169,10,76,105,249,80             <045>
9440 DATA 254,184,173,1,28,149,37,232,224,7,208,
   243,32,151,244,165,22,69,23              <024>
9450 DATA 69,24,69,25,69,26,240,7,198,9,208,192,
   76,30,244,165,24,197,6,240               <014>
9460 DATA 3,76,11,244,133,34,169,6,133,49,76,60,
   4,165,18,166,19,133,22,134               <249>
9470 DATA 23,165,6,133,24,165,7,133,25,169,0,69,
   22,69,23,69,24,69,25,133,26              <059>
9480 DATA 32,52,249,162,90,32,86,245,160,0,80,
   254,184,173,1,28,217,36,0,240            <053>
9490 DATA 6,202,208,237,76,81,245,200,192,8,208,
   234,32,86,245,80,254,184,173             <133>
9500 DATA 1,28,145,48,200,208,245,160,186,80,
   254,184,173,1,28,153,0,1,200,208         <162>
9510 DATA 244,32,224,248,165,56,197,71,240,3,76,
   246,244,32,233,245,197,58,240            <211>
9520 DATA 3,76,2,245,160,0,169,85,32,82,4,185,0,
   6,133,119,44,0,24,16,251,169             <139>
9530 DATA 16,141,0,24,44,0,24,48,251,162,0,138,
   102,119,42,42,102,119,42,42              <020>
9540 DATA 141,0,24,138,102,119,42,42,102,119,42,
   42,141,0,24,138,102,119,42               <028>
9550 DATA 42,102,119,42,42,141,0,24,138,102,119,
   42,42,102,119,42,42,141                  <148>
9560 REM **************************************
   **                                       <156>
9570 DATA 64000,226,25915:REM $FA00-$FAE1   <030>
9580 DATA 0,24,162,2,202,208,253,169,15,141,0,
   24,200,208,173,234,234,234,234           <178>
9590 DATA 234,234,234,173,0,28,9,8,141,0,28,173,
   0,6,208,3,76,158,253,197,24              <169>
9600 DATA 208,249,133,6,173,1,6,133,7,76,101,3,
   133,119,44,0,24,16,251,169,16            <214>
9610 DATA 141,0,24,44,0,24,48,251,162,4,169,0,
   102,119,42,42,102,119,42,42,141          <247>
9620 DATA 0,24,202,208,240,234,234,234,234,234,
   234,169,15,141,0,24,96,96,133            <235>
9630 DATA 0,88,165,0,48,252,120,96,120,234,234,
   234,234,234,234,165,24,141,0             <194>
9640 DATA 6,133,6,165,25,141,1,6,133,7,169,4,
   133,120,169,226,32,130,4,201,2           <187>
9650 DATA 144,51,160,0,132,120,164,120,185,219,
   254,240,18,88,32,118,214,120             <208>
9660 DATA 169,226,32,130,4,201,2,144,26,230,120,
   208,231,169,192,32,130,4,169             <010>
9670 DATA 226,32,130,4,201,2,144,8,169,255,32,
   82,4,76,34,235,173,0,6,240,248           <029>
9680 DATA 197,24,240,196,173,0,6,133,6,173,1,6,
   133,7,76,160,4,234,234,234,234           <095>
9690 DATA 160,0,185,25,244,153              <093>
9700 REM **************************************
   **                                       <040>
9710 DATA 64416,192,24821:REM $FBA0-$FC5F   <203>
9720 DATA 160,89,185,0,2,153,0,3,200,192,0,208,
   245,160,0,177,187,201,36,208             <023>
9730 DATA 3,76,1,245,169,1,133,167,169,0,133,
   144,165,167,32,12,237,169,111            <006>
9740 DATA 32,185,237,165,144,16,11,230,167,165,
   167,201,8,208,230,76,44,247              <023>
9750 DATA 165,167,197,186,240,239,160,0,185,248,
   251,240,6,32,210,255,200,208             <128>
9760 DATA 245,32,225,255,208,251,76,44,247,234,
   234,234,234,13,66,73,84,84,69            <127>
9770 DATA 32,78,85,82,32,70,76,79,80,80,89,32,
   65,78,83,67,72,65,76,84,69,78            <166>
9780 DATA 13,32,67,248,8,72,160,89,185,0,3,153,
   0,2,200,192,0,208,245,166,174            <146>
9790 DATA 164,175,104,40,88,96,255,255,255,255,
   255,255,255,255,255,255,234              <104>
9800 DATA 72,169,244,205,49,3,240,10,141,49,3,
   169,165,141,48,3,104,96,173,76           <185>
9810 DATA 253,141,48,3,173,77,253,141,49,3,104,
   96,141,17,208,76,251,237                 <204>
9820 REM **************************************
   **                                       <161>
9830 DATA 64844,2,412:REM $FD4C-$FD4D       <123>
9840 DATA 184,228                           <126>
10000 DATA 0                                <234>
```

Listing 3. Hypra-Load fest in ein neues Betriebssystem eingebunden (Schluß)

Der C 64 als PET

Wenn Sie CBM 2000, 3000 oder 4000 geschriebene Programme auf Ihrem C 64 laufen lassen wollen, müssen Sie umständlich PEEKs und POKEs ändern. Der »Pet-Simulator« nimmt Ihnen diese Arbeit ab.

Ist das Programm eingegeben und gestartet, werden als erstes die DATAs für das Maschinenprogramm in den Bereich ab Adresse 49152 gePOKEt (SU = Prüfsumme für die Daten). Danach fragt das Programm nach der Zeichenfarbe. Sie werden aufgefordert, eine Zahl zwischen 0 und 15 einzugeben. (0 = schwarz, 1 = weiß,..., 15 = grau 3).

Bei anschließendem Starten des Maschinenprogrammes wird das Basic-ROM in das darunterliegende RAM gePOKEt (Basic-Interpreter kopieren). Anschließend wird das Bildschirm-RAM von Adresse 1024 nach Adresse 32768 verlegt. Basic-Speicheranfang und -ende, werden dem des PET angepaßt. Weiterhin wird in der POKE-Routine des Basic-Interpreters ein Eingriff vorgenommen, nach der der Computer aus dem Interpreter in eine Routine des Maschinenprogramms springt. Hier wird überprüft, ob das Bildschirm-RAM angesprochen wurde. Trifft dies zu, wird die dazugehörige Farb-RAM-Adresse berechnet und der vorher festgelegte Farbwert (Zeichenfarbe) hineingePOKEt. Um das Zurücksetzen des Bildschirmes auf das C 64-Format zu vermeiden (durch Drücken der RUN/STOP- und RESTORE-Tasten) wird die RESTORE-Taste durch Verändern des NMI-Vektors ausgeschaltet. Programme können aber noch mit der RUN/STOP-Taste unterbrochen werden.

Nach Ablauf des Maschinenprogrammes meldet sich der Computer mit »PET-SIMULATOR AKTIV«. Sie können jetzt immer noch die Zeichenfarbe mit POKE 49239, ZF (ZF = Zeichenfarbe — siehe oben) ändern. Wenn Sie jetzt zum Beispiel POKE 32768,1 eingeben, erscheint ein »A« am linken oberen Bildschirmrand in der gewählten Zeichenfarbe. Schlußbemerkung: Bevor Sie das Programm starten, empfiehlt es sich, es vorher abzuspeichern, da sich das Programm selbständig löscht.

(Wolfgang Hopf/rg)

Listing »Pet-Simulator«

```
1 REM ***********************************
2 REM *        PET - SIMULATOR          *
3 REM *                                 *
4 REM *        BY W. HOFF 1984          *
5 REM ***********************************
6 :
7 REM PROGRAMM VOR DEM START ABSPEICHERN
8 :
10 FORI=49152TO49152+91:READA:SU=SU+A
20 POKEI,A:NEXT
30 IFSU<>12552THENEND
40 PRINT"{CLR}{DOWN}{DOWN}BITTE WAEHLEN SIE DIE ZEICHE
NFARBE"
50 PRINT"{DOWN}(0-15 EINGEBEN)!{DOWN}"
60 INPUTZF:IFZF>=0ANDZF<=15THENPOKE49239
,ZF:POKE646,ZF:GOTO80
70 PRINT"{DOWN}NICHT ERLAUBT":FORI=1TO1000:NE
XT:GOTO40
80 SYS49152:POKE1,54:PRINT"{CLR}PET-SIMULATO
R AKTIV":NEW
10000 DATA160,0,132,254,169,160,133,255
10005 DATA177,254,145,254,200,208,249,23
0
10010 DATA255,166,255,224,192,208,241
10015 DATA169,5,141,0,221,141,24,208,169
10020 DATA128,141,136,2,133,56,169,4
10025 DATA133,44,169,0,141,0,4,169
10030 DATA63,141,37,184,169,192,141,38
10035 DATA184,169,193,141,24,3,96,32
10040 DATA235,183,24,165,21,201,128,144
10045 DATA18,201,132,176,14,24,105,88
10050 DATA133,255,165,20,133,254,169,0
10055 DATA234,145,254,96
60000 :
60010 REM LISTE DER VERWENDETEN COMMODOR
E-STEUERZEICHEN
60020 REM "{CLR}" = CLR
60030 REM "{DOWN}" = CRSR-DOWN

READY.
```

Parameterübergabe an Programme in Maschinensprache

Der SYS-Befehl läßt es zu, daß neben der Startadresse des Maschinenprogrammes auch Parameter übergeben werden.

Das kann zum Beispiel so aussehen: SYS adresse,p1,p2,p3,... . Wobei »adresse« Startadresse bedeutet und p1 bis p3 die Parameter sind. Für die Verarbeitung der Parameter ist es wichtig zwischen 1-Byte- und 2-Byte-Parametern zu unterscheiden, da unterschiedliche Interpreterroutinen nötig sind.

Hier sind die benötigten Interpreterroutinen:
Komma = $AEFD
1byte = $B79E
2byte = $B7EB

Dabei ist die Routine »Komma« nötig, um die Kommas vor den Parametern zu erkennnen und die Parameter zu trennen. Anschließend können dann die Parameter mit »1byte« oder »2byte« eingelesen werden.

Übergabe von 1-Byte-Werten

Die Befehle
JSR $AEFD ; Komma
JSR $B79E ; 1byte
laden den Parameter bei SYS...,p1 ins X-Register des Prozessors, wo er für die weitere Verarbeitung bereitsteht. Jetzt ist es zum Beispiel möglich p1 zu verarbeiten und mit JSR Komma : JSR 1byte den nächsten Parameter (p2) zu holen. Angewandt wird dies im Beispielprogramm Cursorsetzen.

Folgendes passiert: Nacheinander werden »zeile« (120,130) und »spalte« (160,170) eingelesen und schließlich verarbeitet. Die Routine $FFF0 setzt bei gelöschtem Carry-Flag den Cursor nach den Werten im X- und Y-Register an die gewünschte Position, wie schon in Ausgabe 7/84 beschrieben.

Aufruf des Hilfsprogrammes das im Kassettenpuffer liegt:
SYS 828,zeile,spalte
(siehe Listing 1)

SYS 828,10,5:PRINT CHR$(42) setzt den Stern * auf Zeile 10, Spalte 5.

Übergabe von 2-Byte-Werten

Die Befehle
JSR $AEFD : Komma
JSR $B7EB : 2byte
bringen den ersten Parameter bei SYS...,p1,p2 auf die Zeropage Adresse $14/$15, und den zweiten ins X-Register, da die Routine $B79E : 1byte (siehe oben) mit aufgerufen wird. Hierbei wird p1 nach Low- und High-Byte getrennt abgespeichert. Sein Wert kann also zwischen 0 und 65535 (= 2↑15) liegen. Der Interpreter benutzt diese Routine, um zum Beispiel bei POKE Speicherzellen mit bestimmten Werten zu laden.

Das nächste kleine Programm soll nun den Gebrauch der Routine 2byte = $B7EB demonstrieren. Es erspart das doppelte POKEn von Bildschirm- und Farb-RAM. Der Aufruf hat die Form:
SYS 828,position, zeichen,farbe

Da »position« hier ein 2-Byte-Wert ist, muß die Routine $B7EB benutzt werden, gleichzeitig wird dadurch »zeichen« ins X-Register geladen. »Farbe« wird dann später in Zeile 210 über $B79E geladen (siehe Listing 2).

SYS828,1063,1,1 setzt ein weißes A in die rechte obere Ecke des Bildschirms.

Zum Schluß noch die Basic-Lader der Demoprogramme, für diejenigen, die keinen Assembler oder Monitor besitzen, aber die Routinen gebrauchen können. Sie liegen jeweils im Kassettenpuffer (ab 828). Durch Verändern der Werte in Zeile 110 lassen sie sich jedoch nach Bedarf verschieben.

(Markus Kuhn/rg)

```
10 REM   ********************          <225>
20 REM   *   MARKUS KUHN      *        <000>
30 REM   *   BAHLENSTR.52     *        <057>
40 REM   *   4 D'DORF 13      *        <057>
50 REM   *   0211/767519      *        <075>
60 REM   *                    *        <031>
70 REM   *  C-64 + VC 1541    *        <146>
80 REM   ********************          <039>
90 :                                   <148>
100 REM BASICLADEPROGRAMM FUER  DEMO 1  CURSORS
   ETZEN                               <015>
110 FOR I = 828 TO 850                 <205>
120 READ X : POKE I,X : S=S+X : NEXT   <069>
130 DATA  32,253,174, 32,158,183,138, 72, 32,
   253,174, 32                         <040>
140 DATA 158,183,138,168,104,170, 24, 32,240,
   255, 96                             <213>
150 IF S <> 3101 THEN PRINT "FEHLER IN DATAS
    !!" : END                          <215>
160 PRINT"OK": END                     <209>
```

Beispiel-Listing 1

```
10 REM   ********************          <225>
20 REM   *   MARKUS KUHN      *        <000>
30 REM   *   BAHLENSTR.52     *        <057>
40 REM   *   4 D'DORF 13      *        <057>
50 REM   *   0211/767519      *        <075>
60 REM   *                    *        <031>
70 REM   *  C-64 + VC 1541    *        <146>
80 REM   ********************          <039>
90 :                                   <148>
100 REM BASICLADEPROGRAMM FUER  DEMO 2  ZEICHEN
   SETZEN                              <056>
110 FOR I = 828 TO 856                 <211>
120 READ X : POKE I,X : S=S+X : NEXT   <069>
130 DATA  32,253,174, 32,235,183,138,160,  0,
   145, 20,165                         <026>
140 DATA  21, 73,220,133, 21, 32,253,174, 32,
   158,183,138                         <036>
150 DATA 160,  0,145, 20, 96           <251>
160 IF S <> 3396 THEN PRINT "FEHLER IN DATAS
    !!" : END                          <241>
170 PRINT "OK" : END                   <219>
```

Beispiel-Listing 2

Große Buchstaben

Wer hat in Titelbildern eigener Spiele nicht schon über die »normalen« Buchstaben geklagt? Mit diesem Programm wird das anders.

Auf die Idee, dieses Programm zu verfassen, kam ich, als mir die immer gleich großen Buchstaben in den Titelbildern vieler Spiele so auf den Wecker gingen, daß ich selbst zum Programmierwerkzeug griff.

Das Programm stellt auf dem Bildschirm maximal 86, in der Höhe doppelt so große, Buchstaben dar. Diese könne jeweils aus 1 bis 2 Farben bestehen.

Nun zum Programm selbst. In Zeile 5 bis 6 stehen die DATAs für das Maschinenprogramm, das den Bereich ab 12288 auf 0 setzt. Dadurch wird ein fließender Aufbau der Buchstaben garantiert. Die Zeilen 7 bis 10 lesen und starten das Maschinen-Programm. In Zeile 20 werden die Grundadresse des VIC sowie die neue Zeichensatz-Startadresse Variablen zugewiesen. Zeile 22 schaltet auf den neuen Zeichensatz um. Zeile 25 dient zum Setzen der Screen-Farben.

Die Variable »D« in Zeile 30 legt fest, ab welchem Zeichen umdefiniert werden kann. Der Wert ist 82, da bis hier alle wichtigen und oft benötigten Zeichen, noch zum Auslesen nötig sind. Ab Zeile 60 beginnt dann das Setzen der umzudefinierenden Zeichen. Von Zeile 120 bis 180 werden dann die gesetzten Zeichen umdefiniert und damit die neue Schrift auf dem Screen dargestellt. Das Ganze funktioniert wie folgt. Zuerst werden die alten Zeichen je viermal Byte für Byte ausgelesen. Jedes einzelne Byte wird dann zweimal dargestellt. Jetzt ist ein Zeichen umdefiniert. Nun wird der zweite Teil des Zeichens ausgelesen, verdoppelt und dem darunterliegenden Zeichen zugeordnet. In dem Unterprogramm ab Zeile 300 wird die Anfangsposition einer Druckzeile festgelegt.

Die DATAs für den Text können dann ab 500 stehen. Wenn dieses Programm aber später nur als Unterprogramm verwendet werden soll, empfiehlt es sich die DATA-Zeile ans Ende dieses Programms zu verlegen.

Eine DATA-Zeile ist folgendermaßen aufgebaut:
1. Spalte in der der Text später beginnen soll
2. Zeile
3. Anzahl der Zeichen die in eine Reihe sollen
4. Farbe obere Zeichenhälfte
5. Farbe untere Hälfte
6. -XX. Hier stehen die Bildschirmcodes der darzustellenden Zeichen.

Wichtig — als letztes Datum muß unbedingt fünfmal die -1 stehen. Da das Programm sehr kurz ist, ist es hervorragend zur Titelbildgestaltung von Spielen, Anwenderprogrammen oder ähnlichem geeignet.

(Matthias Baldauf/rg)

```
0 rem ***********************************
1 rem *    b i g -- l e t t e r s       *
2 rem *         1984 by mkb-soft        *
3 rem * matthias baldauf 06361/7162     *
4 rem *       luitpoldstrasse 62        *
5 rem *       6760 rockenhausen         *
6 rem ***********************************
7 data 134,64,169,0,133,254,162,48,134,2
55,162,0,145,254,200,208,251
8 data 230,255,166,255,224,56,208,241,96
9 fort=52500to52500+25:reada:poket,a:nex
t
10 sys52500: rem * zeichen ram auf 0 dam
it fliesender aufbau
20 v=53248:x=12288:print"ⓢ":rem * x ist
startadresse des neuen zeichensatzes
22 pokev+24,29
25 pokev+32,0:pokev+33,0
30 d=82:rem * bildschirmcode ab dem zeic
hen umdefiniert werden
40 y=x+82*8
55 gosub300
60 fori=1toza:rem * anzahl einzulesender
 buchstaben
70 poke1,55:poke56334,1:q=1
80 forj=0to1:rem * 2 mal da zeichen jetz
t doppelt so hoch
85 ifd>255then400
90 pokec+i+j*40,d:rem * setzen des zeich
ens
100 pokee+i+j*40,fa(q):rem * setzen der
farben
110 q=q+1:d=d+1:next
120 reads
130 poke56334,0:poke1,51
140 forj=0to7:m=peek(53248+(s*8)+j):rem
* auslesen des alten zeichens (klein)
150 fork=0to1:rem * verdoppeln
160 pokey,m:y=y+1:nextk,j,i
170 poke1,55:poke56334,1
180 goto55
300 rem *** anfang setzen ***
301 :
305 readsp,ze,za,fa(1),fa(2):rem spalte,
zeile,anzahl der zeichen,farbe 1,farbe 2
306 ifsp=-1then400
310 c=1024+sp+ze*40:rem * wo text auf sc
reen steht * spalte und zeile*40
320 e=55296+sp+ze*40:rem * dasselbe im f
arbbspeicher
330 return
400 rem *** einlesen beenden ***
401 :
402 end
403 rem ** hier geht spaeter programm we
iter
500 rem *** schrift datas ***
501 rem *** spalte * zeile * anzahl der
zeichen * farbe 1 * farbe 2 * schrift
502 :
503 data 10,4,19,7,10,42,32,20,8,5,32,2,
9,7,32,12,5,20,20,5,18,19,32,42
505 data 17,8,4,8,9,49,57,56,52
510 data 9,11,21,14,6,3,15,16,25,18,9,7,
8,20,32,2,25,32,13,11,2,45,19,15,6,20
520 data 8,19,23,4,4,22,9,5,12,32,19,16,
1,19,19,32,2,5,9,13,32,20,5,19,20
530 data 32,33,33
10000 data-1,-1,-1,-1,-1
```

Restore für Unterprogramme

Wenn beide Programmteile, Hauptprogramm und Unterprogramme, DATAs enthalten, muß sichergestellt werden, daß auch wirklich die richtigen Werte gelesen werden. Wenn man nicht aufpaßt, kann es passieren, daß das Unterprogramm DATAs aus dem Hauptprogramm liest. Wie kann man das verhindern? Es gibt eine umständliche Methode: Man kann eine kleine Basic-Erweiterung einbauen, den RESTORE X-Befehl. Es geht aber auch einfacher. Die Zeropage, das sind die ersten 256 Byte des Speichers, hilft uns bei der Lösung des Problems. Genauer gesagt, die Adressen 65/66 und 122/123. Schlagen wir im C 64-Handbuch auf Seite 162 nach, dann steht dort:

65 - 66 Adresse des aktuellen DATA-Elements
122 - 123 Basic-Zeiger innerhalb der Subroutine

Mit diesen Informationen läßt sich schon etwas anfangen. Wenn das Unterprogramm angesprungen wird, dann sollte der Zeiger in Speicherstelle 122/123 auf die Adresse des Unterprogramms im Speicher stehen. POKEt man diese Werte in die Zeilen 65/55 mit

POKE 65,PEEK(122)
POKE 66,PEEK(123),

so wird beim nächsten READ der Wert gelesen, der hinter dieser Basic-Zeile mit den POKEs steht, also das erste DATA-Element innerhalb des Unterprogramms. Nach dem Rücksprung aus dem Unterprogramm muß der Zeiger eventuell auch im Hauptprogramm wieder gestellt werden.

In dem kurzen Demo-Listing werden drei Unterprogramme in zufälliger Reihenfolge aufgerufen. (Stephan Pätzold/gk)

```
1 REM     ************************      <128>
2 REM     *         DEMO         *      <010>
3 REM     *  SUBROUTINE-RESTORE  *      <071>
4 REM     ************************      <131>
5 PRINT"{CLR,6SPACE}TASTE DRUECKEN !"   <104>
6 PRINT:PRINT                           <114>
10 X=INT(RND(TI)*3)+1                   <125>
20 ON X GOSUB 1000,2000,3000            <040>
25 POKE 65,PEEK(122):POKE 66,PEEK(123)  <001>
30 READ A$:PRINT A$                     <066>
50 DATA " HAUPTPRG."                    <146>
100 GOTO 10                             <078>
1000 REM   *** SUBROUTINE 1 ***         <183>
1005 :                                  <042>
1010 POKE 65,PEEK(122):POKE 66,PEEK(123) <221>
1020 FOR I=1 TO 4:READ A:PRINT A;:NEXT  <145>
1030 READ A$:PRINT A$;                  <105>
1040 POKE 198,0:WAIT 198,1              <116>
1050 DATA 1,11,111,1111,"UP1{2SPACE}"   <085>
1060 RETURN                             <182>
1070 :                                  <108>
2000 REM   *** SUBROUTINE 2 ***         <164>
2005 :                                  <022>
2010 POKE 65,PEEK(122):POKE 66,PEEK(123) <201>
2020 FOR I=1 TO 4:READ A:PRINT A;:NEXT  <125>
2030 READ A$:PRINT A$;                  <084>
2040 POKE 198,0:WAIT 198,1              <095>
2050 DATA 2,22,222,2222,"UP 2 "         <076>
2060 RETURN                             <162>
2070 :                                  <088>
3000 REM   *** SUBROUTINE 3 ***         <145>
3005 :                                  <002>
3010 POKE 65,PEEK(122):POKE 66,PEEK(123) <181>
3020 FOR I=1 TO 4:READ A:PRINT A;:NEXT  <105>
3030 READ A$:PRINT A$;                  <064>
3040 POKE 198,0:WAIT 198,1              <075>
3050 DATA 3,33,333,3333,"UP{2SPACE}3"   <066>
3060 RETURN                             <141>
3070 :                                  <067>
```

Demo-Programm zu RESTORE mit zwei POKEs

Lösungen von Abenteuerspielen

Dallas Quest

Nachdem wir die Lösung von The Blade of Blackpool veröffentlicht hatten, bekamen wir Hunderte von Zuschriften mit der Bitte, dem einen oder anderen bei dem Abenteuerspiel xyz an der Stelle abc weiter zu helfen. Dies ist natürlich aus den verschiedensten Gründen nicht möglich. Dazu müßte bei den meisten Abenteuerspielen die Vorgeschichte bekannt sein, um genau zu wissen, welche Gegenstände man bei sich trägt, welchen Weg man vorher zu dieser Stelle beschritten hat, etc. Ein extrem zeitaufwendiges Unterfangen also. Zudem versteht sich das 64'er Magazin nicht als Spiele-Zeitschrift, obgleich dieser Aspekt des Computereinsatzes nicht außer Acht gelassen wird.

Für 15 bekannte Abenteuerspiele gibt es jetzt übrigens bei Markt & Technik ein Buch mit ausführlicher Beschreibung des Lösungsweges. Da aber ein Buch nicht so aktuell wie eine Zeitschrift sein kann, wollen wir in loser Folge für die neuesten Abenteuerspiele Lösungen anbieten. Unsere Leser sind daher zur regen Mitarbeit aufgerufen. Haben Sie ein Adventure gelöst, so lassen Sie doch Ihre verzweifelten Mitabenteurer daran teilhaben. Den Anfang macht Wolfgang Habich mit der Lösung von Dallas Quest.

east, north, take sunglases, north, give sunglases, go barn, drop owl, take shovel, south, south, south, take envelope, west, take bugle, west, wait, wait, blow bugle, dig, look tombstone, read epitaph, east, drop money, north, open desk, take pouch, north, north, west, west, north, look plane, give envelope, open knapsack, look knapsack, take parachute, drop ring, close knapsack, take knapsack, jump, open pouch, give pouch, close pouch, south, south, look, look parrot, tickle anaconda, south, south, go dinghy, open pouch, give pouch, close pouch, row boat, blow bugle, read sign, go post, open pouch, give pouch, close pouch, look ladder, drop all, open knapsack, (mit take und drop alles einpacken bis auf parachute und bugle),close knapsack, pull curtain, take flashlight, light flashlight, climb ladder, read sign, drop flashlight, east, go post, take knapsack, climb ladder, drop knapsack, east, go post, take shovel, climb ladder, take all, west, unlit flashlight, open knapsack, take photograph, show photograph, read sign, drop photograph, take coconuts, west, look monkey, take pouch, open pouch, give pouch, give eggs, take mirror, give mirror, take ring, wave ring, warm eggs, light flashlight, drop ring, look floor, take map, no, unlit flaslight, give map.

Auf Anweisung des Computers return drücken!
Viel Spaß beim Nachvollziehen.

(Wolfgang Habich)

Programmierwettbewerb: Dokumentationshilfe

1000 Mark zu gewinnen: Schreiben Sie ein Programm, das die Dokumentation automatisch erstellt.

Die Aufgabe, die wir diesmal stellen, ist nicht nur eine Herausforderung an Programmierer, sondern soll zudem für Software-Entwickler ein nützliches Utility sein. Es geht um die Programmierung einer erweiterten Crossreferenzliste. Eine Crossreferenzliste durchsucht per Definition ein beliebiges Programm nach Variablen und Sprungbefehlen und gibt sie auf einem Drucker in gut lesbarer Form aus. Wir wollen aber in diesem Programmierwettbewerb ein vollständiges Werkzeug zur Dokumentation eines sich in der Entwicklung befindlichen oder fertigen Programms erhalten. Im einzelnen sollte das Programm folgendes können:

1. Alle Programmzeilennummern drucken, die Sprünge enthalten. Ausgegeben werden soll die Zeilennummer, dahinter die Zeilen, die angesprungen werden.

2. Ausgabe aller Programmzeilen, die angesprungen werden, wenn möglich mit den Zeilen, von denen aus der Sprung erfolgt.

3. Ausgabe aller im Programm verwendeten Variablen.

3.1 In der Reihenfolge, wie sie im Programm auftauchen.

3.2 In sortierter Reihenfolge: Sortiert nach Gruppe (Integer, Real, Strings und Felder) sowie alphabetisch.

3.3 In welcher Zeile sie definiert werden (Variable =) und in welcher Zeile sie benutzt werden (= Variable).

3.4 Es soll zu jeder Variable ein Kommentar eingegeben werden können.

4. Denkbar wäre auch, die ganze Prozedur innerhalb wählbarer Grenzen (zum Beispiel zwischen Zeile 1000 und 2000) eines Programms ablaufen zu lassen.

Wie Sie aus dem letzten Punkt ersehen können, sind den Ideen keine Grenzen gesetzt. Wichtig ist vor allen Dingen, daß ein komplettes Dokumentationsprogramm für die eigene Entwicklung und zur Analyse fremder Programme zustande kommt. So könnte eine automatische Aufschlüsselung nach Zeilennummern oder die Erstellung eines Fluß- oder Nassi-Shneidermann-Diagramms durchaus mit eingebaut werden. Lassen Sie Ihre Phantasie spielen und dokumentieren eigene und fremde Programme auf die bestmögliche Art und Weise.

Es wird mindestens zwei Gewinner geben: Einer für die beste Lösung in Basic, der andere für das beste Assembler-Programm.

Wenn Ihre Lösung von der oben genannten Aufgabenstellung etwas abweicht, so ist das keine Disqualifikation. Bewertungskriterien werden vor allem sein: Nutzbarkeit, Übersichtlichkeit, Schnelligkeit und Komfort.

Schicken Sie Ihre Lösung unter dem Stichwort »Programmierwettbewerb: Dokumentationshilfe« an folgende Adresse:

Markt & Technik Verlag AG, Redaktion 64'er, Hans-Pinsel-Str. 2, 8013 Haar bei München

Wollen Sie einen gebrauchten Computer verkaufen oder erwerben? Suchen Sie Zubehör? Haben Sie Software anzubieten oder suchen Sie Programme oder Verbindungen? Der COMPUTER-MARKT von »64er« bietet allen Computerfans die Gelegenheit, für nur 5,— DM eine private Kleinanzeige mit bis zu 5 Zeilen Text in der Rubrik Ihrer Wahl aufzugeben. Und so kommt Ihre private Kleinanzeige in den COMPUTER-MARKT der **Februar-Ausgabe** (erscheint am 18. Januar 85): Schicken Sie Ihren Anzeigentext bis zum 20. Dezember 84 (Datum des Poststempels und Anzeigenschluß) an »64er«. Später eingehende Aufträge werden in der **März-Ausgabe** (erscheint am 15. Februar 85) veröffentlicht.

Am besten verwenden Sie dazu die vorbereitete Auftragskarte am Anfang des Heftes. Bitte beachten Sie: Ihr Anzeigentext darf maximal 5 Zeilen mit je 32 Buchstaben betragen. Überweisen Sie den Anzeigenpreis von DM 5,— auf das Postscheckkonto Nr. 14199-803 beim Postscheckamt mit dem Vermerk »Markt & Technik, 64er« oder schicken Sie uns DM 5,— als Scheck oder in Bargeld. Der Verlag behält sich die Veröffentlichung längerer Texte vor. Kleinanzeigen, die entsprechend gekennzeichnet sind, oder deren Text auf eine gewerbliche Tätigkeit schließen läßt, werden in der Rubrik »Gewerbliche Kleinanzeigen« zum Preis von DM 10,— je Zeile Text veröffentlicht.

Private Kleinanzeigen

Private Kleinanzeigen Private Kleinanzeigen Private Kleinanzeigen

COMMODORE 64

C64 in Stereo! Audio-Interface für HiFi-Stereo-Ton, Anschluß von C64 an jeden Verstärker, kpl. 55 DM. Info: G. Zweschper, I.-Kant-Str. 19, 6901 Dossenheim, Tel. 06221/861295

Wait a moment
Suche Hyper Olympic, Champion Boxing, Bruce Lee (nur Tausch). Oliver Pusch, Erlenstr. 88, 4100 Duisburg 1, 0203/776012, Sa.

★ ★ Last-Gutschriftenverwaltung ★ ★ Bis 870 KD/Statist./Datenlistg./Zugriff ü. Kd.-Nr. o. Name/Sort. n. Kd.-Nr. o. Name/Ausdr. EPSON-Comd./Maschinenspr./Tel. 07233/1505

NEUES BETRIEBSSYSTEM BMS-System, farb. Bildschirm, F-Tasten belegt, zahlr. Sonderf., neu. DISKWORKER hilft bei der Diskarbeit. H. Schifferdecker, Tel. 0221/5236 27 o. 7604577

Akustikkoppler für C64, 208 DM, fertig montiert im formgerechten Gehäuse, nur in User-Port stecken, Orig.-Ans. Armin Stockem, Berghausen 13, 5778 Meschede, Tel. 0291/1221

Datasette f. C64/VC 20, DM 70, Basic-Kurskassette kostenl., dazu Bücher 64-intern, DM 25/Multiplan-64 DM 40, 4 Bücher f. VC 20, DM 40/Joyst. DM 15, Spielemod. kostenl. Tel. 02191/67810

★ Relaiskarte m. 8 freipr. Relais ★ für Eisenbahnsteuerung, Lauflicht, Schaltuhr ..., Anzeige ü. 8 LEDs, inkl. Schaltbeispiele, steckfertig DM 90,—. Tel. 07142/41489, 17-18 h

Floppy 1541 ★ 40 Spuren ★ formatieren, lesen, schreiben und kopieren, Anleitung f. 20-DM-Schein an Manfred Wyrwas, Dimhausen 56, 2830 Bassum 1

VERKAUFE
original SIMONs BASIC-MODUL! Besser als die Disk-Version! Modul und Orig.-Handbuch DM 125,—. Uwe Bärtels, Tel. 06173/1536

★ CMB-64 Orig.-Soft- und -Hardware ★ 6-Basic 2 ★ 8 K EPROM + Handb. DM 175, Teleterm + V24-Schnittst., DFÜ DM 95, Maindat dtsch. Dateiverw. + 2 Disk 55. Jäger, Bergw.str. 9, 5630 Remscheid 1

POKE ■ PEEK ■ SYS ■ WAIT ■ TIPS ■ viele Seiten m. Beschreibung 5 DM ■ Dallas-Quest-Lösung 3 DM ■ Vorkasse an ■ Niels Schmidt, Im Hagen 17, 4800 Bielefeld 15

C64, mal keine Raubkopie, aber trotzdem Super-Games, z. B. Warfire Copy, Disksorter, Monitor. Info nur 80 Pf. in Briefmarken an **H. Rek**, Görlitzerstr. 6, 4040 Neuss

NEU NEU NEU NEU NEU NEU NEU Kein Platz für 2 Drucker? PORT-Bus schafft Platz! Info gegen R.-Porto: M. Diefenbach, Weidigweg 19, 61 Darmstadt

Suche Programme aller Art f. C64! Disk oder Kass., Listen an: J. Lenhard, Reinwartstr. 24, 5630 Remscheid 11 ■■■

★ ★ ★ ★ ★ TAUSCHE ★ ★ ★ ★ ★ Programme f. C64, 300 Prog., Liste an od. Tel.: Alexander Krämer, Raise 6, 7930 Ehingen, Tel. 07391/8118

Suche PAINT MAGIC nur auf Kassette. Anruf unter Nummer 07552/8579 ab 14 Uhr

★★★★★★★★★★★★★★★★★★ VERK.-T. eig. Prg., z. B. Vok.spez., Kopierr., Biorhythm. ... Liste: V. Kurz, Heinr.-Rorb.-Weg 19, 706 Schorndorf ★★★★★★★★★★★★★★★★★★

Reset-Schalter f. d. C64 ohne Löten f. 10 DM, Pen Toke f. Fort Apocalypse, 99 Hubschrauber 5 DM. Bei B. D. Klemmer, Reifferscheidstr. 17, 5030 Hürth

VERKAUFE Dallas Quest 30 DM u. Lösung 3 DM, Poke f. 98 Leben bei F. Apok. 3 DM, Strip Poker 2 Mädchen 30 DM, 2 weitere Mädchen 20 DM. Tel. 04191/3788, 19-20 Uhr, n. Disk

Commodore 64 ★★★★★★★★★★ Verkaufe Pokes aller Art! Und Modellbahn. Ruft einfach an: Bernd Gugel, Tel. 09104/2927

★ Computer defekt? Falscher Resettaster? Der einzige m. Schutzschaltung bei mir, 10 DM für VC 64/20 ★ Christian Wöhler ★ Moritzstr. 70 ★ 4300 Essen 1

Verk. C64 + VC1541 + Literatur + 100 Disk. + 2 Joyst. + Fastdisk auf Modul + Centronics-Interf., auch einzeln. André Claaßen, Aldekerkstr. 9, 4000 D'dorf, ab 18 Uhr: 0211/502452

C64 und Drucker 1526/MPS802: Textprogr. m. deutschem Zeichensatz, Disk 35,— DM. A. Schwerd, Adalbertstr. 90, 8000 München 40

Printer VC1520 wegen Systemwechsel zu verkaufen, sehr guter Zustand + Stifte + Papier, VB 290,— DM. W. Willing, Weidenhauser Str. 28, 3550 Marburg. Bitte keine Anrufe!

■ ■ ■ Suche gute Software ■ ■ ■
■ ■ vor allem Karate-Actionprg. und ■ ■
■ was es sonst noch gibt. Anruf od. ■
■ Liste an C. Wodtke, Böttgerstr. 2, ■
■ 5205 St. Augustin 3, ■
■ Telefon 02241/314175 ■

Tausche C64-Programme aller Art • • • • • • • • • • • • • • • • • • Schreibt an W. von Destinon, Langestr. 9, 2840 Diepholz 1 od. Tel. 05441/3558 ab 18 h

■■■■■■■■■■■■■■■■■■■■ Verkaufe C64 + Floppy 1541 + Koala-Painter + ü. 10 Disks f. VB 1350,— DM. Tel. 0991/25538 ab 17 Uhr

Suche folgende Anleitungen gegen neueste Software od. Bares: Superbase 64, Oxford-Pascal, SM Joker 64, Basic Bär, Vizawrite, nur München! Telefon 089/789352, Andreas

Wer brennt mir EPROMS von meinen Pgr.? Suche Monitor und/oder 80 Zeichenkarte gegen beste Software (Spiele u. Anwender) und/oder Bares, nur Raum München, Tel. 789352

• • • • • • • • • • • • • • • • • •
TOP-SOFTWARE
Wird von uns gesucht ★ evtl. Verk. o. Tausch ★ Anrufe o. Listen an M. Bialas ★ Freiburgerstr. 2, 6200 Wsb.-Delkenheim, Tel. 06122/51701

■■■■■■■■■■■■■■■■■■■■ **C64 ■ Spiel des Lebens ■ Das Spiel um Glück, Mißerfolg u. Geld. 38 K! Bis zu 6 Spieler (!) od. allein gegen den Computer** ■■■■■■■■■■■■

■ ■ ■ ■ ■ ■ CAD ■ ■ ■ ■ ■ ■ Suche CAD-Programm od. ähnliches Zeichenprogramm f. den C64. Angebote an Gerhard Gruber, Ulr.-v.-Huttenstr. 6, 8266 Töging

Verkaufe C64-Original-Software, keine Kopien, Nato Commander 39,—, Beach Head 39,—, Basic-Kurs 29,—, auf 1 Disk., Ultisynth. 64 auf Kass. 59,—. Th. Krupp, Sandgrubenweg 13, 6834 Ketsch

C64: Die bewährte Gripsgynamstik (Gesellsch.-Spiele) und Anw.-Prog. (Büro/Verw.) sind bei Gerd Müller, Riedeselstr. 64, 61 DA, 140 erhältl. Orig.-Prg! Info: Freibrief

★ ★ **RESETTASTER** ★ **7 DM** ★ ★ nur Anstecken, kein Löten + **Anl.**, alle getestet und nur **7 DM** an Thomas Gruteser, Maasweg 12, 5144 Wegberg ★

Den Poke f. »Hunchback« gibt es kostenlos: 5704, 50. Über zwanzig (!) weitere (f. Hat Mack, Survivor ...) auf Poke-Liste für nur 10 DM! T. Holzner, Linkstr. 13, 8000 München 45

★ Hallo! Wer hat eine Floppy zu
★ viel? Bezahle mit 2 Topspielen?
★ (Disk. mitschicken) + Adr. an
★ Michael Müller, Am Ludwigsberg 13,
★ 6607 Quierschied

ACHTUNG!
Verkaufe Original-Datasette, 1 Jahr alt, nur DM 75,—. Th. Maurer, Am weißen Rain 63, 6292 Weilmünster, Telefon 06472/7942

★ ★ ★ ★ Hilfe, ich sterbe ★ ★ ★ ★ 1541! Suche schon ewig lebendige 1541. Biete 400,—, außerdem engl. Vokabelpr. m. Vokabeln + Bücher. Ruft bitte an. Tel. 06087/332

■ ■ ■ ■ ■ ENDLICH! ■ ■ ■ ■ ■ Modem-Gehäuse f. Bausätze. **K. Schubert, Rottmannstr. 10, 6900 Heidelberg**, 48,50 per NN inkl. Porto u. Verp.

C64, Super-Adventure Teufelshof, Ziel des Spieles ist es, einen Schatz zu entwenden. 20 DM an Georg Köster, Hermann-Ehlers-Str. 8, 2903 Ofen ☐

★ ★ ★ ★ Das geht alle an ★ ★ ★ ★ Suche Lernprogramme und Tauschpartner! Nur Kassette! Schickt Eure Angebote an Andreas Chwatal, Gebr.-Lang-Str. 17c, 636 Friedberg

Wer schreibt Biorhythmusprogr., geschr. f. TRS-80, Mod. 1 m. Line-Printer VII (ähnl. Seikosha) für C64 + MPS 801 gegen gutes Geld um. G. Ullrich, Tel. 05156/8448

C64 gesucht mit Datasette o. Diskettenlaufwerk m. Büchern u. Progr. zu verkaufen. ZX81 m. 16 K u. Buch. G. Wimmer, Eschweiler Str. 5, 5110 Alsdorf, Tel. 02404/61933

• • • • • • • • • • • • • • • • • •
Suche für C64 Comal-Version 0.14. Tel. 05424/5916
• • • • • • • • • • • • • • • • • •

Suche neue Software (Hyper Sports) auf Diskette C64.
Günter Wild, Streitbergstr. 87, 8000 München 60 ★★★★★★★★★★

Software ★ ★ ★ ★ Software
Ich tausche meine Programme (Kass.) geg. Ihre! O. BRZOSKA, Telefon 04191/6227

An alle Besitzer von Z80, 80Z und Floppyparallelbuskarten. Suche (Belohnung!) techn. Unterlagen zum Selbstbau. Michael Schunk, J.-Ringelnatz-Str. 3, 2190 Cuxhaven

★ ★ Suche Floppy VC 1541 ★ ★
★ Guter Zustand, etwa 1 Jahr, ★
★ zahle bis 450 DM ★
★ Ralf Ruth, Neunkirchen/Saar ★
★ ★ Tel. 06821/31868 ★ ★

Tausche Software auf Kassette. Interessantes Material vorhanden. Schickst Du mir Deine Liste, schicke ich Dir meine. HORST KREBS, HANSE-LUCKE 5, 6101 WERSAU

Verkaufe 80-Zeichenkarte (Jann), Platzerweiterung u. CP/M-Modul, Telefon 06221/13443

Armer Schüler sucht defekten C64 für 50,— DM. Volker Brill, Büschfelderstr. 6, 6612 Schmelz 3, Tel. 06887/3862

■ Kaufe defekte VC 20-, 64-Floppys,
■ Drucker, Steckk. usw. Anfragen mit
■ mit Zustands- u. Fehlerbeschr. an
■ Thomas Stepputat, Eisenbahnstr. 55a
■ 6580 Idar-Oberstein

★ ★ ★ ★ ★ DRINGEND ★ ★ ★ ★ ★ Suche gebrauchten C64 + Datasette (auch einzeln). Telefon 0761/273332 ab 18 Uhr

★★★★★★★★★★★★★★★★★★
POKES
für fast alle Games wie z. B. FORT Jump. Jr. usw. gegen geringen Preis. Info unter Tel. 069/581103 od. 573878

Achtung:

Wir machen unsere Inserenten darauf aufmerksam, daß das **Angebot**, der **Verkauf** oder die **Verbreitung** von urheberrechtlich geschützter Software nur für Originalprogramme erlaubt ist.

Das Herstellen, Anbieten, Verkaufen und Verbreiten von **»Raubkopien«** verstößt gegen das Urheberrechtsgesetz und kann straf- und zivilrechtlich verfolgt werden. Bei Verstößen muß mit Anwalts- und Gerichtskosten von über DM 1000,— gerechnet werden.

Originalprogramme sind am Copyright-Hinweis und am Originalaufkleber des Datenträgers (Diskette oder Kassette) zu erkennen und normalerweise originalverpackt. Mit dem Kauf von Raubkopien erwirbt der Käufer auch kein Eigentum und geht das Risiko einer jederzeitigen Beschlagnahmung ein.

Wir bitten unsere Leser in deren eigenem Interesse, Raubkopien von Original-Software weder anzubieten, zu verkaufen noch zu verbreiten. Erziehungsberechtigte haften für ihre Kinder.

Der Verlag wird in Zukunft keine Anzeigen mehr veröffentlichen, die darauf schließen lassen, daß Raubkopien angeboten werden.

COMPUTER-MARKT

Private Kleinanzeigen Private Kleinanzeigen

Suche Software auf Kassette f. C64. Prog.-Tausch wird bevorzugt. Listen an/von: **Thorsten Roßkamp, Ostlandstr. 6, 2935 Bockhorn 1**

★★★★★★★★★★★★★★★★★
Suche C64 + Software + Hardware
Tel. 069/737924 ab 19 Uhr
★★★★★★★★★★★★★★★★★

Suche gute 64'er-Literatur! Besonders Data Becker: Peeks & Pokes sowie das Maschinensprachebuch. Angebote an Carsten Grantz, Mörikestr. 11, 2190 Cuxhaven 13

Suche Summergames, Hyper Olympics und andere Sportspiele, außerdem Castle Wolfenstein auf Kassette. Angebote (Preis VB) an D. Hartwig, Marienfelder Allee 178, 1000 Berlin 48

■■■■■ SUCHE ■■■■■
Commodore 64 + Floppy (bis 600 DM) oder Commodore 64 + Datasette (bis 600 DM). ■ R. Welte, Seestr. 63, 798 Ravensburg, Tel. 0751/16923 ■

Suche C64 mit Datasette od. Floppy, auch mit Zubehör, möglichst günstig da Schüler. Stephan Stöbe, Jägerstr. 36c, 2842 Lohne, Tel. 04442/6064 ab 17 Uhr

Nicht wegwerfen — Nicht wegwerfen
□ Defekte VC64- u. VC 20-Peripherie □
□ Datasette, Floppy, Drucker usw. ge- □
□ sucht. Schriftl. Angeb. an Kaminsky, □
□ Benediktstr. 2, 469 Herne 2 □

Wer hat Lernprogramme für mich, z. B. Deutsch/Englisch u. a.? Nur auf Kassette. Tel. 05361/72619, auch Tausch mit Spiele

! SUCHE !
Gebrauchten C64 + Datasette!
Nehme günstigstes Angebot!
Raimund Busch, Engeserlandstr. 272, 5450 Neuwied 1, **Tel. 02631/28216**

Verk. wegen Hobbyaufgabe C64 + VC1541 + Seikosha GP-100 VC + Software (45 Disk.), Preis n. VB, auch einzeln abzugeben. T. 05523/2493 ab ca. 17h

Suche Bauanleitung für DIN-A4-Plotter mit Interface an VC 64. WER hat schon einen Plotter f. VC 64 selbst gebaut? Jürgen KOCH, Niebuhrstr. 1, 5 Köln 41

•••••••••••••••••
SUCHE FLOPPY 1521!
Angebote bis DM 400,— an
Reinhard Schorer, Heinrichstr. 1,
8950 Kaufbeuren
•••••••••••••••••

■ VERKAUFE ■
1 Oxford-Pascal (orig.verp.) + engl. + deutsch. Handbuch, 1 Turbo-Disk-Modul, 5- bis 6mal schnellere Floppy
Tel. 02174/3153

SUCHE
SIMONs BASIC-Modulversion
Verkaufe DATA BECKER PASCAL 6/84, Version 3.1/3, Original 60,— DM, Telefon 02225/12470

LOAD ERROR LOAD ERROR LOAD
Einstell- u. Reperaturanleitungen. Floppy u. Datasette. Jetzt rep. auch Laien selbst! DM 10 + adr. Rückumschl., J. Renfer, Ebertstr. 43, 67 Ludwigshafen

•••••••••••••••••
Rossmöller-EPROMMER inkl. Disk. od. Tape m. Software **229,— DM.** Tel. 02236/47518 ab 17 Uhr
•••••••••••••••••

C 64
Lightpen Bs.: 45,—/fertig: 65,— m. Funktionsgarantie + Software + Anl. + Beschreibung! Gleich bestellen tägl. v. 17-21 Uhr. Tel. 0261/63296

★ ★ Suche CP-80X-Hardcopy ★ ★ gegen Programme etc. Angebote und Anfragen an Nils Rix, Heerwieserweg 10, 5880 Lüdenscheid

TAUSCHE: Super-8-Ausrüstung gegen Floppy od. Farbmonitor
SUCHE: Turbo-Tape-Steckmodul u. Spiel-Prog. auf Kass. f. 64. Jäger, Richard-Wagner-Str. 81, 66 Saarbrücken 3

Suche Summer-Games auf Kass., zahle bis zu 10 DM. Angebot an Boris Schreiner, Blumenstr. 20, 7052 Schwaikheim, Tel. 07195/52651

Software gesucht! Ich suche selbstgeschr. Programme, Tips, Tricks usw. für den C64. Selbstverständlich gegen Honorar! W. Möhle, Tel. 0511/5352126

•••••••••••••••••
ACHTUNG! C64!
Commodore 64 — Simons-Basic
Orig.-Disk Tel. 06202/13200
•••••••••••••••••

Suche preisgünstiges Diskettenlaufwerk VC1541. Mathias Bachmann, Schillerstr. 19, 7745 Schonach
•••••••••••••••••

Gebr. C64 gesucht
Außerdem alle ADVENTURES
Tel. 06136/43028

Kaufe Ultima 2 + 3, Realm. of Impossibility, C. Boxing, Tom, Millionär, The Catacombs, Montezuma, Kaiser, Bruce Lee, auch Tausch. C. Rothenhöfer, Hopgarten 1, 3181 Grafhorst ■■■■■■■■■■

★ ★ 64-Club, RAUM Osnabrück ★ ★ sucht weitere Mitglieder. Clubzeitung und Info f. 3 DM (bei Beitritt Verrechnung). U. Henke, Kretescher Turm 6, 45 Osnabrück

■ C64 + 1541 + Datasette (alles 5 Mon. alt) + 2 Joyst. + **200** Diskseiten mit ca. **900** Top-Prog., kompl. 2150,— DM. J. Spindelmann, Opladener Str. 30, 4018 Langenfeld

COMMODORE-Drucker **1526/MPS-802,** deutsche Zeichensätze für **SM-TEXT** o. **VIZAWRITE** u. **C64.** Telefon 040/8501556

Suche Top-Sport-, Action- u. Adventuregames auf Kass.! Liste m. Preisen an Michael Grüber, Brunnenstr. 8A, 4450 Lingen/Ems ★★★★★★★★★★★★★

Suche **dringend** C64, evtl. auch Floppy VC1541, bis 900 DM! Meldungen an Olaf Lüdemann, Willenbrockstr. 28, 2723 Scheessel, Tel. 04263/771

FORTRAN ★ Suche einen Fortran IV o. Fortran V-Compiler f. d. C64. Angebote an H. J. Kuckertz, Im Hag 34, 5180 Eschweiler, Tel. 02403/28151 ab 19 Uhr ★★★★★★★★★★★★★★

★ SUCHE MATRIX-DRUCKER ★
In Frage kommen Einschl.-Interf. EPSON RX80 u. FX80 od. ähnliche Qualitätsdrucker, Mo-Do ab 18 Uhr, Tel. 040/2207953 ★★★★★★★★★★

Tausche oder verkaufe Fernsteuerung Terratop + Starion + SEA Commander + Pegasus 3 + Schütze-Schiffe, 1500,—, Tausch gegen VC1541 od. Farbmonitor od. Drucker, ab MPS801. Telefon 06626/8966

C64-Schaltinterface f. User-Port m. Stecker, 8 Ausg., einz. ansteuerb., Anz. über 8 LEDs, Trafo + 8 Relais, 220 V, 2000 W/Aus. + Anl., DM 90, F. Huber, Bietigheimer Str. 18, 7120 Bietigheim

MÖCHTE EINSTEIGEN! Suche C64, Floppy, Drucker, Monitor sowie Software, nehme bestes Angebot, M. Baumann, Alvensstr. 20, 1000 Berlin 30

SUCHE Anleitungen zu Hulk, Dallas, Hobbit, Cauch Flights II usw.). TAUSCHE Software. Andreas Eberhart, Mayenerstr. 32/2, 705 Waiblingen. **Zahle gut!**

Suche Bedienungsanleitungen f. jedes Software-Prog., deutsch o. englisch. Bitte Liste mit Preisvorst. an P. Cruse, Postfach 605354, D-2000 Hamburg 60

Suche Softw. f. C64 jeder Art, insbes. S.A.M., nur Disk., auch Tausch mögl., Liste an P. Tannert, Bottnerstr. 1, 7141 Schwieberdg. Tel. 07150/41893 tags.

COMMODORE C64 UND ZUBEHÖR GESUCHT
Telefon 0651/16366

Versierter Berater (Schüler etc.) f. C64-Probleme im Bereich 4178 Kevelaer gegen Entgelt dringend gesucht! Angebote bitte an Postfach 108, 4178 Kevelaer 1

★ ★ ★ ÖSTERREICH ★ ★ ★
Suche Tauschpartner/in C64. Liste an ROBERT HIEBL, Klaubling 36, A-3350 HAAG

★ ★ Mein C64 braucht Futter ★ ★
Suche billigst (auf Disk.), **heiße Games** und **Fantasy-Spiele** m. **Anleitung**. Listen an St. Frenzel, Vogelgartenstr. 29, 7332 Eislingen

Bankkaufmann sucht Bank- und andere Software. Umfangreiche Progr.-Bibliothek vorhanden. Möglichst Disk-Tausch. Tel. 07256/8397, werktags ab 17 Uhr (außer SCHLADO)

★ Wer hilft mir beim Umbau des Druckers Shinwa CP80 auf CP80X mit Rat oder/und TAT (?EPROM?)? Suche Vizastar (Hallo Snoopie) + Logo-Anleitung. Tel. 07246/1340

C64 + Floppy 1541 + Monitor + 15 Disks + Programmierhandbuch (alles originalverpackt) f. 1580 DM zu verkaufen. Tel. 0621/815450

C64 — — — — — — — — — — C64
Suche Floppy 1541 <350 od. gegen Spielprogr. Tel. 00411/7154445 (CH)
C64 — — — — — — — — — — C64

Suche Anleitungen für C64-Programme. Simons Basic, Exbasic, Adventures, Spiele usw., nur deutsch gegen Unkostenbeitrag. Bitte an Gerald Poiger, Vorgartenstr. 45/2/4/13, 1200 Wien

Suche Progr. aller Art f. VC64 gegen Bez. (per NN) od. Tausch-Angebote od. Listen an Gerald Poiger, Vorgartenstr. 45/2/4/13, 1200 Wien/Österreich. Tel. 35-70-193

Hallo **C64-Anwender**! Programmlücke? Ich schließe sie! Mit spottbilligen Progr., Liste anfordern bei T. Wings, Brunsbergweg 5B, 5810 Witten 2 ■■■■■■■

• • • • • • • • • • • • • • • • • •
ÖSTERREICH
Tauschpartner gesucht! Liste! 1 Disk voll mit Utilities nur öS 250 (inkl. Disk). Reinhard Mayer, Erdbergstr. 97/11, 1030 Wien/A

Drucker Brother HR5C f. C64, direkt anschließbar mit Netzgerät u. Zubehör u. Garantie, NP DM 540,— f. DM 400,— zu verkaufen, Tel. 02161/641224 abends

TRS-80-Pocketcomputer, Basic, 1,9 K, 24stell. Anzeige + Drucke, 16 Z/Zeile m. Akku, Rekorderanschl., kompl. m. Ladegerät, Handb., Kabel, Finanzprogr., zus. 250,—. Tel. 0221/3601274 ab 19 Uhr

ACHTUNG! Verkaufe wegen Systemwechsel C64 + 1541 (m. Garantie), ROOS 80 Zeichenk. u. Parallelbus f. 1541 (6x schneller). M. Becker, Tel. 0221/621910

■■■■■■■■■■■■■■■■■■■■■
Verk. ZX-Spectrum, ½ Jahr, mit 6 Spielprgr. u. 5 Fachb., Volker Mack, Eichenstr. 50, 4700 Hamm 1
■■■■■■■■■■■■■■■■■■■■■

UMSONST will ich nichts, aber eine gebr., evt. defekte Floppy, die billigste wird gekauft von Oliver Schindelbeck, Dieboldsberg 26, 7297 Alpirsbach

□ □ □ WER KANN HELFEN! □ □ □
Suche aus Chip Nr. 8, lauffähiges Simul-X-Programm f. C64. Angebot an Georg Kastler, Erlenring 18, 5559 Hetzerath, nach 18 Uhr 06508/7901

Anfänger sucht Programme aller Art f. C64. Listen an Georg Kastler, Erlenring 18, 5559 Hetzerath/Mosel, Tel. 06508/7901 nach 18 Uhr

Suche Anl. aller Art! Angebot m. P. (Orig./Kopie), bes. Superbase, Flight II, Vizawrite! Wer hat Erfahr. m. Viza + PRIV 3000 (Centr.) an C64? R. Bremer, Natz-Thier-Str. 6, 4421 Reken

Neuwertiger VC64 + Floppy + Dataset. + 100 Diskettenseiten an Meistbietenden ab 1300 DM! Angebote an Stefan Müller, Tel. 0209/1612723 ab 20 Uhr

Verkaufe (wegen Systemwechsel) Oxford Pascal, 120 DM, Original-Disk. + Handbuch. Tel. 0721/373569 (ev. öfters probieren)

SUCHE TAUSCHPARTNER
Tausche alle Arten von Programmen auf Kassette. Liste gleich mitschicken. Peter Brill Remmesweilerstr. 7, 6694 Urexweiler

ADVENTURE-FREUNDE AUFGEPASST!
Mäusekönig-Super-Grafik ★ starker Sound ★ nur auf Diskette, da etliche Teile f. nur DM 20,— bei Daniel Maier, Tel. 0821/605483 ★ Friedberg ★

C64 + Floppy + Datasette-Turbomdoul + Seikosha 100A + Interface + 4 Data Becker-Bücher wegen Aufgabe zu verk. VB 2100 DM. Tel. 0761/499838 ab 18 Uhr

SUCHE Anleitungen (mögl. deutsch) zu Flight II, SM-TEXT 64, Koala-Paint, Necromancer, THE LAST ONE usw. ★ Angebote an U. Schwenke de Wall jun., Harzweg 8, 338 Goslar 1

Suche folg., gut erhalt. DATA BECKER-Bücher: Adventures, Lexikon zum C64, Floppy-Buch, Grafikbuch z. C64 ★ Angebote an U. Schwenke de Wall jun., Harzweg 8, 338 Goslar 1

Suche gebrauchten Commodore 64, Angebote (Alter- u. Preisangabe) an M. Kober, Dorfstr. 3a, 6483 Bad Soden-Ahl

Verkaufe VC 64, Floppy 1541, MPS 801, Joystick (Comp.-Prgr.), Literatur, 150 Prog. für nur 1950 DM. Ludger Brägelmann, Spannskamp 26, Zim. 716, 2000 Hamburg 54, Tel. 040/545213

Suche äußerst billig Programme f. C64 jeglicher Art — Sprachen, Spielpr., Grafik usw. Liste an Markus Klövekorn, Nordstr. 21, 4400 Münster, Tel. 0251/28009

Suche C64 m. Floppy f. 450 DM, auch mit kleinen Fehlern. Angebot an Alwin Süßen, Ringstr. 104, 2963 Südbrookmerland

Suche billige, gute Software C64 auf Kass. (Spiele, Videothek, Progr.), Floppy VC1541. Angebote u. Listen an Dieter Jensen, Rastherr-Wieting-Str. 11, 2880 Brake

SUCHE COMPUTERSCHROTT
Defekte Computer, Floppys usw.
Zahle angemessene Preise!!
Tel. 0441/511151 nach 19 Uhr

Private Kleinanzeigen

★ ★ Progr. f. Turbo-Tape zu lang? ★ ★ Ext. Turbo-Tape saved bis zu 184 Disketten-Blocks! (Archon, Jammin), f. nur 10,— bei Thomas Wenzel, Dresdenerstr. 20, 3250 Hameln 1

Verkaufe Tandy TRS-80 Color + Datasette + 2 Joystick, VB 300 DM ★ **Suche** VC 64, m. Floppy 1541 ★ Bitte melden bei Bernd Kütting, Tel. 0231/597293. **EILT!**

Student sucht möglichst billig DATA BECKER PASCAL V3.0 und/oder OXFORD PASCAL (m. Anleitungen). Angebote an Marcus Gießen, Gleiwitzer Str. 14, 4048 Grevenbroich 5

Hallo C64-Fans! Ich gründe einen C64-Club. Schreibt mir! Rückporto! (Für Action wird gesorgt.) Jochen Rauscher, Hermann-Hesse-Str. 5, 7430 Metzingen-Neuhausen

OPA + ENKEL suchen Progr., besonders Spiele zum Tauschen, schickt Eure Listen. Unkosten, auch Porto, werden voll vergütet. **M. Kockel, Lindenstr. 8, 6097 Trebur 1**

C64 ★ sorgfältig getestete Progr. (z. B. Raumlehre/Geometrie/Chemie) gegen geringen Unkostenbeitrag abzugeben. Info anfordern bei P. Stein, Lupinenstr. 29, 4040 Neuss

C64 + 1541 + MPS 801 + 2 Joyst. + Superbase 64 + Kontomat + Faktumat + Datamat + Floppy-Buch + Disk. C64 Lex. 64, Tips + Tricks + 64 für Einst., C64 Basic + Comal + allg. Ausg. 64'er Magaz. DM 2750. Tel. 06221/20209

Verkaufe C64 + Floppy + viel Software (20 Disks) + Kass.-Interface f. nur 3000,— DM. Alles nur 8 Monate alt! Info bei Martin Elsner, Buschdorferstr. 25, 5300 Bonn 1

Drucker MPS 801, 9 Mon., DM 480, einschl. Kabel f. VC 64. Friedrich Schäfer, Auerweg 3, 5042 Erftstadt 24, Telefon 02235/77753

Suche Vizastar, Calc Result u. a., Centronics-Schnittst. (Hardware), 80 Zeichen, Z80-Karte. Hans Münstermann, Postfach 1234, 4952 Porta Westfalica

C64 ■ Darl.- u. Hypothek.-Tilgungspl., 3 Sonder-Tilg.! 1 Zinswechs.! 1 Guthaben! Aut. Laufzeiterr.! Ausg. Druck. o. Schirm gegen Scheck o. Schein! DM 30. mit Disk! Gessner, Graefestr. 33, 35 Kassel

■ ■ ■ ACHTUNG LEHRER ■ ■ ■ SCHÜLERNOTENVERWALTUNG mit CBM64, 3 Fächer m. je 4 Teilgebieten. INFO geg. Freiumschlag bei Hans Schmidt, Bahnhofstr. 19, 6798 Kusel

SUCHE
C64-Software auf Diskette
Angebote an THOMAS UNGER,
UHLANDSTR. 6, 7434 RIEDERICH

Suche alle Karate-Programme, die es für den Commodore 64 gibt (z. B. Bruce Lee u. Karateka)! Ich kaufe oder tausche die Programme. Tel. 07161/82261

Suche Software für C64: Makro-Assembler, Basic-Compiler, ExDos & Disk-Doktor. Angebote an M. Grebler, Westendstr. 18a, 8016 Feldkirchen, Tel. 089/9033074

Menügesteuert durch die Welt der ASCII-Code. Sehr übers. m. fast allen ASCII-Coden. Auf Disk od. Kass., für nur 20,— DM. VK bei Detlef Link, Piwipperstr. 17, 4047 Dormagen

● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
Weihnachtsgeschenk
CBM 64, originalverpackt, mit Garantie, Einzelstück, f. 590 DM abzugeben. Benecke, Tel. 06332/3976

Verkaufe CBM64 + Floppy 1541 + 2 Joysticks + Paddle + 30 Disketten m. Progr. + Bücher, VB 1300,— DM. Liste von Ch. Bernhofer, Hochreit 10, 8221 St. Georgen

Suche gebrauchten C64 od. VC 20 m. Datasette od. Floppy u. Software FIBU o. ä., nur schriftliche Angebote an F. Kies, Loensstr. 4, 3170 Gifhorn

Verkaufe KW-Empfänger SONY ICF 2001. **Suche** VC1541-Floppy. Anfrage u. Angebot: H. Gassmann, Knipferstr. 14, 8830 Treuchtlingen

EPSON FX-80-Drucker-Interface
von Data Becker zu kaufen gesucht. Verkaufe C64, SX64, Epson PX-8, Akustikkoppler, Disketten u. andere Hardware. Tel. 05252/2172

HALLO LEHRER! Schulverwaltung! Auf dem C64 (Zeugnisse, Klassen-/Kurslisten, Schulstat.). Info gegen 2 DM bei W. Müting, Siemensstr. 4, 4780 Lippstadt

● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
CBM-4032-Gehäuse zu verkaufen.
Tel. 089/758077
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●

Verkaufe folgende Originalspiele: Hero: Kass. 30 DM, Decathlon: Kass. 20 DM, Poleposition: Modul 40 DM. Suche und kaufe: Hyper Sports u. Hyper O. auf Disk. Tel. 04392/1805

Suche f. C64 Anwendungsprgr., möglichst m. Beschreibung u. gute Spiele, Angebote an BRIMI, 2381 Schaalby

Verkaufe C64 m. Datasette, 3 Mon., gebr., wegen Neukauf zum Sonderpreis von nur 675,— DM. Info geg. Rückporto. W. Hirschmann, Postfach 86, 8399 Griesbach

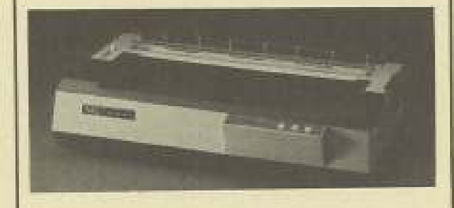
Suche billig Floppy 1541, Software auf Kassette. Gunter Wild, Streitbergstr. 87, 8000 München 60, Tel. 089/873606

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Orig. Simons Basic Disk 80 DM, Tauschpartner gesucht! An- u. Verkauf v. 64er-Prg. Info bei PLK 51502, 29 Oldenburg. Suche Modem od. Akustikkoppler

Raum Köln: Verkaufe C64 m. Datasette, Joystick u. viele Spiele auf Kass. (z. B. Hobbit, ZAXXON). VB 700 DM. Norbert Rath, Cheruskerstr. 1, 5000 Köln 21, Tel. 817597

VC-64 — VC-64 — VC-64 — VC-64
Verkaufe Originalprogramme Solo-Flight, ISM-64, Forth-64, Originalsoftware **sehr billig.** Tel. 08193/6387

★ Suche für C64 guten Assembler, Flugkarten für Flight II u. F-15! **Andreas Schreyer, Flurlgasse 7, 8440 Straubing, Tel. 09421/10688** ★ ★ ★ ★ ★ ★



Private Kleinanzeigen

Hobbyaufläg.: Verk. C64 + Floppy 1541 + el. Schreibm. Privileg 3000 m. Interface, Typenräder, Datasette, Software (150 Disk. doppelseitig voll), Literatur, 43 DM. Tel. 0761/74497

C64 + 1541 + 1526 + 1530 + Textomat + Datamat (neu) + Kontomat + 3 Joystick + ca. 20 Disketten für DM 2100,—. Bürger, Neuhausen 31, 8673 Rehau, Tel. 09287/3870

Verkaufe C64 mit Floppy 1541, 3 Monate alt. T. Mayer, Güntertstr. 20, CH-4310 Rheinfelden, Tel. 061/875065

Suche Programme aller Art f. Datasette/Floppy. Andreas Gerr, Karl-Streit-Str. 2, 8730 Bad Kissingen, Tel. 0971/4559

Sprachausgabe für C64, bestückte Platine, betriebsfertig, nur in User-Port stecken, plus Software f. Sprachausgabe, DM 170,— G. Conrad, Tel. 040/892110

★ ★ ★ BMC BX-80 ★ ★ ★
Neuwertiger Drucker mit Interface f. C64 zu verkaufen. Preis: 1200,— DM (VB). Tel. 089/3154185 ab 17 Uhr ★

VC-64 + Floppy + Datasette + Farbmonitor (1701) + Drucker (MPS 801) + 3 Joysticks + Literatur + 350 Programme etc. zu verkaufen, Komplettpreis 3500 DM. Tel. 02107/3370

Suche für 64! Suche für 64! Suche für 64! Diskettenlocher unter 15 DM und auf Diskette: Hypra-Load, Disk Dump, Smony. Diskettenorganisation. Melden bei Sven, Tel. 030/4152079

• • • • • • • • • • • • • • • • • •
Verkaufe »Brush up your English«. Alle drei Teile für nur 110 DM. M. Müller, Bergstr. 6, 8572 Auerbach, Tel. 09643/1285 ab 18 Uhr

Verk. Videocass., Vorspann, 7 Eingaben möglich, werd. umrahmt ausgedruckt, mit Uhrzeit. Korr.-Nachtr. mögl. Preis 20 DM (Disk). U. Fink, Oberdischingerstr. 3, 7939 Öpfingen

Turbodiskmodul f. C64 ohne RAM-Verlust & Directory & Fehlerkanal 8000 u. C000, frei schaltbar, auch als Turbo-Tape. R. Weisang, Butterpfad 14, 6682 Ottweiler 4, Tel. 06858/556

Verkaufe MPS-801-Drucker wegen Systemwechsel f. nur 500 DM. Ralph Pankow, Weimarer Str. 35, 8503 Altdorf, Tel. 09187/8210

Suche gebrauchten Farbportable- bzw. Monitor, auch defekt, außerdem Platine u. Plan f. Sprachausgabe C64 ges. Angebote an Josef Dolpp, Vöhringerweg 6, 7941 Unlingen

Suche Sportprogramme aller Art (auf Kass.), Summer Games, Football Manager, Boxen, Handball o.ä. Angebot o. Tausch. Jörg Wilmes, Eschensiepen 6, 5600 Wuppertal 2

Schüler sucht Floppy! Bis 350 DM (auch defekt bis 200 DM). Andy Raith, Wolfsbergerstr. 33, 8522 Herzogenaurach

Typenrad-Schreibmasch. Olivetti m. eingeb. Commodore-Interface f. VC 20/64, nur 3 Monate alt, Preis DM 860,—. Tel. 040/6302841 ab 19 Uhr

Suche 64'er-Zeitschrift, Ausgabe 4/84 bis 9/84, Programm Textomat + Kontomat + Floppy C 1541. Tel. 0821/402053

64'er!!! ACHTUNG! Ich schreibe die auf Ihr Problem abgestimmte **Usersoftware**, Basic u. Maschinenspr. N. Teller, B.-u.-Suttner-Str. 4, 2900 Oldenburg, Tel. 0441/54791

■■■■■■■■■■■■■■■■■■■■
7500 Offenburg
Suche VC-64-User in od. um Offenburg zwecks Erfahrungsaustausch. Ruft doch mal an. Tel. 0781/57345

HURRA! Der **HOBBIT** ist gelöst! Ausführliche dt. Beschreibung des komplexen Lösungsweges für 10 DM bar bei C. Elfering, Dimker Allee 54, 4270 Dorsten 11. **HURRA !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

■■■■■■■■■■■■■■■■■■■■
Suche C64-Datasette, alle möglichen Monitore (kompl.), 80-Zeichenkarte, auch einzeln. Tel. 06039/42415

Suche für C64 gute Programme auf Floppy 1541 (Spiele, Anwender-Prg.). Kauf oder Tausch. Listen an Thomas Steiner, Fenkernweg 5, CH-6010 Kriens

Suche Commodore-Floppy 1541 (möglichst nicht älter als 6 Monate) u. einen Matrixdrucker f. den C64. TEl. 05341/36113

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Pascal 64, orig. Data Becker, 70 DM, 1x benutzt. Frank Pischel, Bahnstr. 43, 4220 Dinslaken
★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★

1541 FLOPPY 1541
Suche Floppy 1541, zahle je nach Zustand bis 500 DM, Floppy muß technisch o.k. sein. Angebote an M. Neubauer, Tel. 02605/3537 nach 17 Uhr

SUCHE ANLEITUNGEN. Original oder gute Kopien, evtuell Tausch-Kopierer vorhanden. Floppy günstig gesucht. F. Wanner, Fraunattstr. 39/2, CH-4410 Liestal

Allein macht's keinen Spaß! Suche Anschluß an Club, bin an neuer Clubgründung auch interessiert. Austausch von Programmen. K. D. Dischner, Duisburg, Tel. 371410

Superpreis: Simons Basic mit Demo und Handbuch, Grandmaster, 80-Zeichen-Karte + 5 Topspiele nur DM 50,—. Tel. 0621/708195 ab 17 Uhr oder Wochenende

C64 + Disk + Seikosha-GP-100-VC-Drucker + DX Ford Pascal + Mal 64 Assembler + Simons Basic + Exbasic Level # + 10 Data-Becker-Bücher + 3 Commodore-Bücher, Neuwert 2800,— f. 1900,— VB. Tel. 0511/324840

Ausführliche deutsche Anleitung für Flightsimulator II (40 Seiten, einschließl. Flugkarten) f. 20-DM-Schein bei V. Jungbluth, Platterstr. 14, 6200 Wiesbaden

1 Doppelfloppy Micropower, NP 2798,—, DM VB 2200,—, 1 Monat alt, EPROM-Karte f. 6 x 2764, umschaltbar A000-BFFF + 8000-9FFF 75 DM, 1IR-Fernbed.-Sender + Empfänger 1500 DM, Tel. 0231/252348

Neues aus Vaterstetten:

Mit dem iwt-Programm auf die Zukunft programmiert!



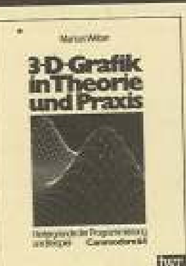
Grafikprogramme werden ›gehirngerecht‹ aufbereitet. Neue Art des Formats – man bekommt ein ›Bild‹ des Befehls, Demo-Programme unterstützen das Gedächtnis, Bildschirm-Hardcopies als schnelles Nachschlagewerk, farbige Übersichtskarten zur Programmier-Erleichterung.
208 S. Spiralh. DM 44,–/ Fr. 44,–



Die Programmierung des Video Interface Chips 6567 ist Hauptthema des Buches. Basic - Grafikprogramme werden von Maschinenprogrammen zum Punkt-/Linienzeichnen unterstützt, was die Schnelligkeit vielfach erhöht, teilweise Basic-Programme direkt in Maschinensprache parallel dargestellt.
152 S./Spiralh. DM 38,–/ Fr. 38,–



Bekanntlich verfügt der C 64 ›von Haus aus‹ über einen Baustein, der die Erzeugung von mehrstimmiger Musik erlaubt. Sowohl der Anfänger ohne musikalische Vorkenntnisse wird angesprochen, als auch der Musiker, der seine Ideen mit Hilfe des Computers umsetzen möchte.
312 Seiten. Spiralh. DM 48,–/Fr. 48,–



Dieses Buch zeigt, wie sich komplizierte Operationen verständlich beschreiben lassen. Es wird demonstriert, wie einfach sich dreidimensionale Probleme lösen lassen. Die Beispiele reichen von der Geraden über das Dreikörperproblem bis hin zum dreidimensionalen Planetensystem.
208 Seiten. Kart. DM 44,–/Fr. 44,–



Einführung auf dem Dragon 32/64 anhand einzelner Routinen in 6809-Maschinensprache, daher auch für andere 6809-Systeme geeignet. Assembler, Disassembler und Fließkommapaket gehören zu den Programmen. Hilfreiches Nachschlagewerk durch ausführliche Befehlsbeschreibungen u. Tabellen.
286 Seiten. Spiralh. DM 44,–/Fr. 44,–



Dieses Buch enthält eine ganze Reihe von sofort lauffähigen Spiel- und Simulationsprogrammen, möchte aber auch dazu anregen, diese Programme zu verändern und weiterzuentwickeln. Besonders reizvoll dürfte es wohl sein, den ›lernenden‹ Programmen noch etwas mehr ›Intelligenz‹ zu verleihen.
208 S./Spiralh. DM 38,–/ Fr. 38,–

Ich bin neugierig auf Ihr Gesamtprogramm! Senden Sie mir umgehend

- ☐ Ihren neuesten Computer- und Elektronik-Literaturkatalog.
- ☐ Ich interessiere mich für Ihre ROBOTIK-Idee.
- ☐ Erbitte Unterlagen über Ihr umfangreiches Software-Programm.
- ☐ Ich möchte mit D.A.T.A BOOKS Zeit und Geld sparen.

MICRO-COMPUTER '85 Halle 4.0, Stand 866

Name/Vorname ______

Firma ______

Abt. ______

Straße/Hausnr. ______

PLZ/Ort ______

64

IWT Verlag, Vaterstetten
Der Fachverlag für Information, Wissenschaft, Technologie



Dahlienstraße 4, 8011 Baldham, Tel. (0 81 06) 3 10 17, Tx 5213989 iwt
Auslieferung Schweiz: Thali AG, Buchhandlung und Verlag, CH-6285 Hitzkirch, Tel. (041) 85 28 28
Auslieferung Österreich: Oberösterreichischer Landesverlag Linz, Fachbuchabteilung, Landstr. 41, A-4010 Linz, Tel. (07 32) 27 81 21/296/245, Tx 02/1014

Wenn mal was nicht funktioniert...

... an Ihrem Commodore Computer
... rufen Sie uns ungeniert –
die Service-Profis von »Rat und Tat«.
Wir warten und reparieren
schnell · preiswert · gut

commodore

- VC 20
- C 64

und die Peripherie

- Floppy-Disk
- Drucker
- Datasette



Bundesweit – an 70 Standorten:

Augsburg-Stadtbergen, Tel. (08 21) 52 40 16 - 17
Berlin, Tel. (0 30) 6 84 60 57 - 59
Bielefeld, Tel. (05 21) 2 08 04 40
Bocholt, Tel. (0 28 71) 18 21 95
Braunschweig, Tel. (05 31) 4 46 71 / 84 50 99
Bremen, Tel. (04 21) 41 43 50
Bremerhaven, Tel. (04 71) 4 91 68
Celle-Altencelle, Tel. (0 51 41) 8 12 30
Cuxhaven, Tel. (0 47 21) 5 15 40 / 3 70 33
Darmstadt, Tel. (0 61 51) 10 92 52
Deggendorf, Tel. (09 91) 3 32 92
Dortmund-Holzwickede, Tel. (0 23 01) 8 74 15 - 16
Düsseldorf-Eller, Tel. (02 11) 21 30 45 / 22 29 58 - 59
Essen-Vogelheim, Tel. (02 01) 3 59 23 - 27
Frankfurt/Main, Tel. (0 69) 41 60 11 - 13
Freiburg-Gundelfingen, Tel. (07 61) 5 88 01 - 02
Fulda-Petersberg, Tel. (06 61) 6 19 10
Gießen, Tel. (06 41) 59 44 - 45
Göttingen, Tel. (05 51) 5 59 40 / 40 92 70
Goslar-Baßgeige, Tel. (0 53 21) 5 05 31 / 5 03 45
Hamburg 74, Tel. (0 40) 73 16 65 - 69
Hannover 1, Tel. (05 11) 3 10 46 39 / 32 77 55 - 56
Heilbronn, Tel. (0 71 31) 4 49 32
Hof/Saale, Tel. (0 92 81) 99 41
Idar-Oberstein, Tel. (0 67 81) 2 78 00
Ingolstadt, Tel. (08 41) 5 80 80
Iserlohn, Tel. (0 23 71) 2 41 51
Kaiserslautern, Tel. (06 31) 85 92 58
Karlsruhe, Tel. (07 21) 13 72 57
Kassel, Tel. (05 61) 7 89 52 51 / 10 31 01
Kempten, Tel. (08 31) 2 41 10
Kiel, Tel. (04 31) 68 00 40
Koblenz-Lützel, Tel. (02 61) 8 20 44 - 45
Köln-Rodenkirchen, Tel. (0 22 36) 6 75 11
Krefeld, Tel. (0 21 51) 75 11 24 / 75 11 83
Landshut, Tel. (08 71) 2 67 60
Limburg/Lahn, Tel. (0 64 31) 2 57 06
Lübeck 1, Tel. (04 51) 89 80 40
Lüneburg, Tel. (0 41 31) 3 66 86
Mannheim, Tel. (06 21) 1 68 33 30
Memmingen, Tel. (0 83 31) 43 35
Minden, Tel. (05 71) 2 80 25 - 26 / 88 12 49
Mönchengladbach-Rheydt, Tel. (0 21 66) 42 08 80
München-Eching, Tel. (0 81 65) 7 42 56 - 57 / 40 80
Münster, Tel. (02 51) 62 40 10
Neumünster, Tel. (0 43 21) 4 20 61 - 68
Neu-Ulm, Tel. (07 31) 8 40 70
Nürnberg-Eibach, Tel. (09 11) 21 38 16 - 18 / 63 20 02
Passau, Tel. (08 51) 5 21 77
Pforzheim, Tel. (0 72 31) 2 40 21 - 22
Ravensburg, Tel. (07 51) 2 51 16
Recklinghausen, Tel. (0 23 61) 20 95 51 / 37 22 79
Regensburg, Tel. (09 41) 5 34 46
Rendsburg, Tel. (0 43 31) 20 43 04
Rosenheim, Tel. (0 80 31) 4 22 05
Saarbrücken, Tel. (06 81) 3 01 72 78 / 3 70 93
Siegen/Haiger, Tel. (0 27 73) 24 46
Singen, Tel. (0 77 31) 6 78 70
Solingen 1, Tel. (0 21 22) 20 08 80
Stuttgart-Leonberg, Tel. (0 71 52) 7 22 38 - 39
Trier, Tel. (06 51) 7 32 09
Villingen, Tel. (0 77 21) 5 41 90
Wiesbaden-Delkenheim, Tel. (0 61 22) 5 22 71 - 72
Wilhelmshaven, Tel. (0 44 21) 4 23 99
Würzburg, Tel. (09 31) 5 02 89

Private Kleinanzeigen

Suche Serviceleitung f. VC-64 u. Floppy 1541 sowie Schaltbild. Suche defekte Floppy 1541 zu günstigem Preis. Dirk Dreesbach, Königsbergerstr. 72, 6239 Kriftel

Data-Becker-Interface f. Epson RX80 DM 185, Textomat, Datamat, Supergrafik, Profimat, Diskomat, Faktumat je DM 75, Original m. Handbuch. Tel. 021 07/24 24

Roßmöller EPROM-Programmiergerät (Testsieger s. 64'er-Ausgabe August) kpl. DM 225 (Selbstkosten), Rombox DM 52 (o. EPROM), Centronics-Kabel RX80/ Userport DM 50. Tel. 021 07/24 24

■■■■■■■■■■■■■■■■■■■■ Suche gebrauchten Commodore-Monitor ★ VC-1702 ★ Jörg Stanke, Moorstr. 26, 4544 Ladbergen, Tel. 054 84/568 ■■■■■■■■■■■■■

★ . WEGEN SYSTEMWECHSEL ★ Floppy 1541 f. 540,— DM + 86 Disk voll Games + Anw.-Disk = 25,—, ab 10 Disk = 20,—, alle = 999,— A. Musahl, Brommberger Str. 97, 28 HB 21, Telefon 04 21/6 16 11 47

Verk. VC-64 + VC 1541 + VC 1526 + Joyst. + 17 Disk. m. ca. 100 Progr. (Magic-Desk, Simons Basic + Datamat + Spiele usw.) + 6 Bücher f. nur 2400,— DM (Wert neu 4400,—). T. Fischer, Tel. 074 52/18 10

Suche Turbo-Tape-Modul u. tausche CBS Colecovision + 2 Topspiele gegen einen C64 kompatiblen Matrix-Drucker, kaufe auch Spiele-Anleitungen ob original od. kopiert. Tel. 026 81/61 37

Kaufe Macrofire, gleichw. Makroassembler (guter Editor), evtl. m. Anleitung, C64-Literatur (Floppy, Grafik, Assembler). Christoph Meinel, Taunusstr. 80, 6233 Kelkheim

C64: Suche Software (Spiele), z. B. ZAXXON, Summer Games, Dimension X, Forbidden Forest ... Nehme fast alles. Listen an J. Benning, Kölner Str. 3, 429 Bocholt

Commodore-Drucker MPS 801, 598,—, MPS 802 nur 748,—, Star Delta 10 zum Superpreis von 1198,—, Taxan-Farbmonitor Vision EX f. 798,—, alles fabrikneu, Tel. 07 21/29 48 4

Orig. Simons Basic (Disk) mit Handbuch 100 DM (per NN)
★ ★ ★ Softwaretausch (nur Disk) ★ ★ ★
Liste an: Achim Schmitt
Werkloh 27, 4600 Dortmund 15

Datenübertragung !!!
Erfahrungsaustausch, Tips, Tricks.
Wer schreibt mir?
W. Krusche, Dorfstr. 5, 8110 Murnau

C64 + Datass. + Joyst. + Neuer Print./Plott. + 10 Papierr. + 12 Farbst. + Basicb. + Kass. + 4 Christiani-Lehrb. + 5 Spielc. (NP ca. 2000 DM), kompl. DM 900,—. Voigt, Kaarster Str. 230, 4040 Neuß 1

64-TOOLKIT 20 BASIC-Befehle Z.B. Disk-BASIC, F-Tastenbelegung sowie RENUM, AUTO, DUMP u.v.a. Für 35 DM (NN) auf Disk abzugeben. R. Hauke, Selkestr. 28, 1 Berlin 44

C 64-Einsteiger sucht Software aller Art (Tape oder Diskette). J. Hutmacher, Hammerstr. 1, 5500 Trier

Verkaufe: C 64+Datasette+VC 1541 sowie Spielprogramm Soccer fast neu! Angebote an S. Sommerer, 8580 Bayreuth, Dr.-Würzburgerstr. 13, Tel. 09 21/ 417 48

C 64-EINSTEIGER
Suche Adventures aller Art auf Kassette!
Listen an:
F. Herrmann, Landvogt-Waserstr. 43, 8405 Winterthur (Schweiz)

Suche auf Disk: Encounter, Zaga, Bruce Lee, Hes Games, Pitfall, Hulk Pitfall 2, Dallas Quest, Flight 2, Pharaos Curse■ Listen an Stefan Brandl■Haus 25½■ 8493 Kötzting

Günstig! C64, Datasette, Monitor, **Akustikkoppler**, MPS-801, MPS-802, **RS232-Schnittstelle**, Disketten, Farbmonitor, Anrufbeantworter zu verk.! Anruf lohnt. Tel. 052 52/21 72

Suche dringend Floppy VC 1541, zahle bis 400,— DM, außerdem Strategie-Spiele gesucht. Angebote an Stefan Rauterkus, Im Hain 2, 6233 Kelkheim/Ts., Tel. 061 95/34 09, Mo. 19-20 Uhr

Akustikkoppler LEX-11 orig. USA von Lexikon, voll-/halbduplex, Schnittstelle RS232, Batterie- od. Netzbetrieb, 300 Baud, neu 498 DM f. 300,— zu verk. Tel. 071 61/8 24 24

Suche Textverarbeitungsprog. Finanzbuchh. und andere Anwenderprog. preiswert. Suche defekte 1541-Floppy. Suche preiswertes Gehäuse von Commod. 30/40er. Tel. 022 45/36 73

Original DATA BECKER-PROGRAMME! Textomat, Pascal 64, Profimat je 45,- DM (Neupreis 99,-), Uni-Tab 30,- DM (NP 69,-), alle Prg. mit Handbuch. 05 11/ 51 48 80 ab 18 Uhr

Hypertape (opt. + akust. Kontrolle) + Supertape (bis 50 KByte) 10 x schneller auf Kassette! Pro Prg. 12 DM im Umschlag an K. Winneckens, Hammarskjöld Str. 43, 2800 Bremen 61

SUCHE C 64 bis 350 DM oder Tausch gegen VC 20 + 150 DM.
Thorsten Mehlich, Oerlinghauserstr. 37, 4902 Bad-Salzuflen-1

C 64 Modemprg. 100% Maschinenspr., 60 K frei für Daten ★ Up-Down-Load ★ Disk- u. Druckfunktionen ★ DM 33,— Verrechnungsscheck——Marten Güttner, Wengerfeld 4, 8116 Eschenlohe

■■■ HIRES 1526/802 ■■■ Drucken v. Koala-, Doodle-, Dia- u. Paint-Magic-Bild. CBM 1526/802, 3 verschiedene Bildgrößen
Prg.-Preis 20,- DM, Tel. 051 93/22 86

Verkaufe meinen IEEE-488-Bus für CBM 64. Umschaltbar, kostet keinen Speicherplatz, läuft mit fast allen Prg's, VB DM 190,-, Tel. 02 21/55 85 38

Verk. Orig. Calc-Result-Anl. 40,— Verbindungskabel C 64/1541 (+Netz, Kabel 1541 10,—) ★ Tausche C 64-Disk-Spiele z.B. 7 Cities of Gold. Liste an: C.J.A., Mozartstr. 6, 8898 Sob

Dringend gesucht: Floppy VC 1541 ■■ und/oder Farbmonitor. Bedingung: komplett in Ordnung. Zahle sofort bar. Angebote an: Malte Grubitzsch, Tel. 02 11/63 12 51 (abends) ■■■■■■■

★ ★ ★ ★ **Anfänger** ★ ★ ★ ★
AN ALLE, die ihren 64 neu bekommen haben!! Schreibt mir: Habe Softw. Mario Bernecker, Alte Poststr. 54, 5551 Monzelfeld, 065 31/36 48

Für unseren Computerclub (C 64, TI 99, Apple) suchen wir noch Mitglieder ab 18 Jahre im Raum Dinslaken und Umgebung, Adresse: Tel. 021 34/9 53 78 ab 16.00 Uhr wochentags

LIGHTPEN zum direkten Zeichnen auf Bildschirm: 2,6 m Kabel, mit Demo + Anl., NP 79,- DM für VB 50,- DM. ★ ★ Suche außerdem TRACKBALL — M. Schäfer, Ringsheimer Weg 37, 535 Euskirchen 25

★ ★ ★ An alle Anfänger ★ ★ ★
Na, habt Ihr noch keine Software? Schreibt mir mal und gebt Euer Alter an: Marco+Kropf, Cusanustr. 17, 5550 Bernkastel, Tel.: 065 31/66 24

★ ★ Suche gebr. Floppy 1541 zum ★ ★ günstigen Preis! Nehme bestes Angebot. Angebote an: M. Rosenbohm, Taubenring 12, 2859 Nordholz, Tel.: 047 41/ 78 47

Verkaufe: Farbdrucker Seikosha GP 700A + Data Becker Interface fast neuw. DM 1100,- (VB) sowie Paint-Pic (Data Becker) orig. mit Disk. v. Handbuch DM 60. 030/691 54 63

VERKAUFE COMMODORE 2001 mit Monitor Datasette und Software für Anfänger und Sammler, suche 1541. Ernst-Peter Eull, Stormstr. 5, 5000 Köln 50, Tel. 022 36/6 26 05

Wegen Hobbyaufgabe verkaufe ich meinen C 64 + Floppy + Datasette + 2 Joysticks + 140 Disks (ca. 1000 Programme) + Erweiterung + Literatur gegen Höchstgebot (VB: 2500 DM). Alles in Topzustand. 062 01/5 35 58

Wer schenkt Schüler defekte Floppy oder Drucker ? Suche auch CBM 64-Besitzer im Raum Passau. Probst Robert, Sandbergstr. 12, 8390 Passau, Tel. 08 51/4 15 12 ab 19.00 Uhr

Suche OXFORD **PASCAL**, KMMM-**PASCAL, PASCAL** 64 V3.0, TURBO-**PASCAL**, HES-**FORTH** — möglichst mit Anleitung. Angebote an Gerhard Schindele, Marienstr. 24, 7512 Rheinstetten 3

Suche Turnier-Programm für Schachturniere im Schweizer System mit 200 Teilnehmern und exaktem Farbwechsel nach jeder Runde. G.G. Rauschenbusch, 070 66/73 46

Commodore-Original-Software zum halben NP. Simons-Basic 100,- DM, Adressen 64 50,- DM, Originale! Keine Raubkopie! Beide zusammen 120 DM. W. Kaupp, Oberwaldstr. 61, 75 KA 41

Suche Vokabeltrainer für norwegisch. Bitte melden bei PUC »Pionier USER Club« c/o Junge Pioniere, Postfach 13 02 03, 4600 Dortmund 1, Tel. 02 31/ 2 75 76

Suche: Amateurfunk-Pgr.: RTTY Oscar 10 neuer QTH-Kenner UKW-Kontestlog Entfernungsber. nur Disk
G. John, 7920 Heidenheim, Hans-Thomastr. 6, Tel. 073 21/6 31 30

■ C 64 Aktien-Depotverwaltung ■
■ Umfangreiche Analysen auf ■
■ Monitor + Drucker; Hires-Grafik ■
■ B. Kremeyer, Meisenpfad 14 ■
■ 4900 Herford, Info kostenlos ■

Verkaufe: Akustikk. Epson CX-21+Interface+SW 600,— DM, Video-Digitizer+ SW 500,— DM, Syntymat+SM Mae 64+ Assemblerk. Asem4 für 150,— DM. Komplett 1000,— DM: Tel. 02 21/87 68 27

★ Suche ★ Suche ★ Suche ★ Suche ★ defekte Floppy VC 1541 (bis 50 DM). Angebote aus dem Großraum Lübeck an: Hartmut Schmidt, Hauptstr. 33, 2407 Bad Schwartau

Suche Drucker u.a. Peripherie für C 64. Dietmar Klug, Steinweg 3 B, 3014 Laatzen 3

Private Kleinanzeigen

Disketten **Nashua 1D** günstig zu verkaufen, Personal Computer **PX-8 von Epson** DM 2300,—, Kontakt zu Computer-Clubs zwecks Diskeinkauf gesucht. Tel.: 05252-2172

Verk. **Commodore 64 + Floppy-Disk** 1541 (noch **Garantie) + Datasette + 20 Disketten (200 Spiele)** für DM 1200.- R. Kraska, Plaggenweg 15, 465 Gelsenkirchen, Tel.: 0209/594230

★★★★ Suche Oxford-Pascal ★★★★
★ Angebote (nur Originale) an ★
★ Wolfgang Uttendorfer ★
★ Eichenweg 21, 7122 Besigheim ★
★★★ 07143/36683 ab 18.00 ★★★

★★★ EPSON-Drucker an C 64 ★★★
Epson FX/RX80 an C 64 Centronics-Schnittstelle Wiesemann Typ 9200 mit Anleitung, VB 190,—, 18-21 Uhr Tel. 05541/4710

C 64: Gesucht werden:
Anwenderprogramme (D oder K)
Zahle nur in bar
Patrick Krojer, Hasenberg
CH-9205 Waldkirch/SG

Suche Software, Tausch oder Kauf (Anwender-Lernprogr., eventuell auch gute Spiele; nur Kassette). Harald Strub, Korellengarten 32, 6550 Bad Kreuznach, T. 0671/71451

★★★ C 64 ★★★ C 64 ★★★
Tausche Software, nur Disk !!!
Verk. Skizzenzeichner DM 20.
Mehr Info bei Anruf, J. Hetz
Tel. 06471/30547 (nach 19.00 Uhr!)

■■■■■■■■■■■■■■■■■■■■■
Tausche 800 Spitzenprogramme ■■
gegen gebr. Farbmonitor ■
Alexander Haas • Römerstr. 35 ■
7460 Balingen 12 (07433) 34267 ■

★★★★★ Achtung Wien ★★★★★
Tausche oder kaufe Software und Zubehör, Drucker, Floppy, Joysticks uvam. Ziegelhofstr. 32-34, 13/1/8, 1220 Wien, Friedrich Kratky C 64

Commodore 64, Drucker 1526, Floppy 1541, mit sehr viel Zubehör (Software), erst 4 x gebraucht, wegen Zeitmangel zu verkaufen, Preis: 2300,— DM VHS, Heike Reiser, Ziegelofenweg 3, 6720 Speyer

LISTINGS
Drucke schnell + zuverlässig. Nur 20 Pf/DIN A. Disk+Geld+frank. Rückumschlag an: B. Meißner, Gmünderstr. 76, 7060 Schorndorf

C 64-Einsteiger sucht Programme, Anschluß an Clubs, Telefon Nr. sowie sonstige Hilfe. Zuschriften an R. Hausmann, Postfach 1232, 4322 Sprockhövel

■■■ Suche für C 64 gebr. o. ■■■
■ def. VC 1541 u. Drucker zum ■
■ fairen Preis!!! Softw.-Tausch ■
■ an: U. Benke, Graslitzer Str. 6 ■
■ 8756 Kahl, 06188/2740 ab 14 Uhr ■

Wer finanziert mir den Ausbau der 64-Anlage durch Kauf v. Büchern usw. Liste gegen Rückporto bei H. Halsch, Fellenweg, 6411 Künzell 3, Tel.: 0661/34671 ab 17.00 Uhr

Suche C 64 + Datasette auch einzeln, evt. Software, Bücher, Joyst. Preis VB Angebote an Jochen Kuhn, Schulweg 13, 8708 Gerbrunn

Astrologie-Horoskop-Kartenlegen für C 64+VC 1541+MPS801 gesucht. Nur Originalware. Angebote mit Probedruck an: U. Umlawski, Von-Sandt-Platz 1, 5000 Köln 21

Suche Anleitung für Star League Baseball von Gamestar — Suche Auflösung für Heroes of Karn — Angebote an Martin Rödig, Brentanostr. 51, 1000 Berlin 41

■■ Bau-Ingenieur-Student sucht: ■■
Fachprogramme **JEDER** Art: Statik, Wasser-/Verkehrswesen, CAD, etc.; ebenso CP/M, FORTRAN. Angebote an R. Hoppe, Am Alten Saupark 9, 3 H 61

★★★★ **Achtung 64er-Fans** ★★★★
Verkaufe komm. ROM-Listing 15,- M. Schumacher, Hinter den Obergärten 28, 6000 Frankfurt 60, Tel.: 06109/22790

Verkaufe VC 64+Floppy 1541+Philips Monitor+AtariTrak-Ball, mit Software u. Literatur, für nur 1500,- DM. Conradt, 3330 Helmstedt, Stendaler Str. 9, 05351/2239

Suche immer neue Software und zuverlässigen Tauschpartner! Suche auch Modem oder Akukoppler!
Angebote an Matthias Kühnel, Grobenbreden 19a, 3977 Marienmünster

C 64 C 64 FAST-TAPE C 64 C 64
DATASETTE wird 10 x schneller
Nur 5,- DM!!!! auf Kassette gelf. Brief an Friedrich Abel, Pf. 111, 7320 Göppingen, Tel. 07161/74009

———> Verkaufe günstig: <———
(wegen Sammlungsauflösung)
ca. 480 Science-Fiction-Romane
(v. a. Perry Rhodan)
Postfach 300548/7000 Stgt.-30

Commodore 64 gesucht — auch defekt
R. Schriewersmann, Talstr. 29
5067 Weiden/Kürten
Tel. 02268/6961

Neu! Spielstandverwaltung (für alle Arten v. Spielen); 2-6 Spieler! Berücksicht. Contra-Re u. Bockrunde sowie Anz. d. Spiele! Disk od. Kass. 15.— + Porto, Tel. 02268/6961

VC 64, Floppy, Datasette, 16er Block, 2 Joyst., Spiele, Progr., Literatur, FFS-AV-Kabel, Reset VB 1400.—
Tel.: 08151/7026 od. 12017

Suche C 64 mit Handbuch und Datasette Peter Kleen, Suchsdorfer Weg 10 C, 2300 Kronshagen, Tel. 0431/581762 (möglichst noch sehr neu)

Ab 19 Uhr Verkaufe alles! Ab 19 Uhr Commodore C 64, Floppy 1541, Joyst., Super-Programme (Top-Spiele, Lehr- u. Nutzprogr.), Literatur, Zubehör
VB 1.800,— DM, Tel. 02651/42418

★★★ EX-DOS & DISK-DOCTOR ★★★
Verkaufe meine Original-Diskette inkl. Handbuch für DM 69.-.
M. Hoppmanns, St. Antonstr. 90, 4150 Krefeld 1

■■■■■■■■■■■■■■■■■■■■■
Neues Betriebssystem für C 64!!!
Mit 6 x schnellerem Laden/belegte F-Tasten/Floppydirektbefehle und vieles mehr für nur 79,- DM !!!
Nähere Informationen bei:
T. Bölting, Kohlstraße 12, 4030 Ratingen 6, Tel.: 02102/67545 anrufen ab 19 h oder Sa. und So.
■■■■■■■■■■■■■■■■■■■■■

★★★★★★ LÖSUNGEN ★★★★★★
von Dallas Quest und Aztec Tomb je 5 DM + Rückporto. Schreibt an S. Sipeer, Joh.-Strauß-Str. 2, 7114 Pfedelbach

DATA 64, C 64-Club, PB-500, 5625 Kallern, Schweiz: 210 Mitglieder aus
★★★★★★★★★★★★★★★★★
der ganzen Schweiz! Probe-Clubzeitschrift gegen Fr. 2.-: Briefmarken

C 64 Disk Tool! Einzelnes Formatieren der Spuren 1-**40!** Erzeugen der Errors 20,21,22,23,27 und 29. Info 80 Pf. K. Raczek, Wickrathberger-12, 5140 Erkelenz, 02431/3236

C 64 Suche Tauschpartner! C64
Habe 350 Programme Kass. u. Disk
M. Martensen, Osterhusumerstr. 65, 2250 Husum, Tel. 04841/73717

Suche gute Programme und Hardware für C 64. Angebote an Rainer Wallenfels, Neuselsbrunn 14, 8500 Nürnberg 50, Tel. 0911/862090

Texto-, Data-, Konto-, Faktumat, Simons Basic, Basic-Kurs, Adressen 64, Datasette 1530, Drucker 1525 25% unter Neupreis, alles Originale, auch einzeln; ab 21.00 h 08361/3609

Adventure-Spieler werden ihre Freude haben, regieren Sie Ihr Volk (1 b. 6 Sp.). Versuchen Sie (C 64) König zu werden. 10 DM für Kass. M. Schmidt, Bornschlade 16, 5204 Lohmar 21

VC 64! Kleine Computergruppe aus Mainz! Wir suchen ständig gebr. und neue Hardware gegen Bezahlung. Ebenso Literatur über VC 64. Tel.: 06131/674889 ab 18 Uhr

Haben Sie mit Ihrem Heimcomputer Platz Probleme? Verkaufe Computertisch (75 x 100 x 60 cm). Information von H.-U. Wodtke, Gartenstr. 8, 3509 Spangenberg 1

★★★★ **COMMODORE 64** ★★★★
★ Suche gute Software. Tel. ★
★ 02841-504152 (ab 17.00 Uhr) ★
★★★★★★★★★★★★★★★★★

Suche billig Flight Simulator 2+Anleitung, The Blade of Blackpool, Bruce Lee Kass. Markus Jürries, 3300 Braunschweig, Köslinstr. 17

WICHTIG!!! Sie erhalten eine Liste aller C 64-Benutzer Ihres PLZ-Gebietes + Ihre Adresse wird gespeichert für DM 20 bar! Von F. Bind, Debusw. 2, 6240 Königstein

COMPUTER-MARKT

Private Kleinanzeigen

BRD ★★★ Österreich ★★★ Schweiz
Suche faire und verläßlich C 64-Partner.
Suche Deu Vizawrite + Star
Listen an A-8680 Mürzzuschlag
Jelinek, Joh.Waldand. 5, 03852/3370

C 64-GRAFOLOGIE
Aus der Handschrift die Charaktereigenschaften ermitteln. Disk 49.-/Kass. 39.-/ Info Kurt Hummel, Brühlstr. 11, 6209 Aarbergen 2

C 64 (neuw.)+Datasette+Prog.Ref. Guide+Masch.Spr.—Ass.+Masch.Spr.—Buch DM 650 ★★★ alle 64'er Ausg. Preis VB ★★★ Esser, Tel. 089/ 688818 ab 18 Uhr

Suche gebr. oder def. 1541 Floppy, Drucker sowie Software u. Anleitungen für C 64
A. Schmit, A-1020, Taborstraße 34/10, Wien, Austria

★★★★★ On-Line Service ★★★★★
Neuer Software-Club, Telefonservice, neueste Software !!!
Sofort Info anfordern — kostenlos
R. Müller, Pf 151502, 8000 München 15

Superpreis: Fasttape, Simons-Basic mit Demo und Handbuch, Grandmaster, 80 Zeichen + 5 Topspiele nur DM 50,—, Tel.: 0621/708195 ab 17 Uhr oder Wochenende.

★★ Kein Drucker? Kein Problem! ★★
★ Drucke Listings Ihrer Programs ★
★u. Hardcopys v. Disk/30 Pf/BL+VP★
★ Copyright garantiert! W. Wiesner ★
4132 Kamp-Lintf. Monterkampweg 41

Hallo Adventure-Freaks: Habt Ihr Probleme mit The Dallas Quest? Gegen 5,- DM als Schein erhaltet Ihr die Lösung bei St. Artmann, Danzigerstr. 43, 3472 Beverungen 1

■ **Experimentierplatine C 64** ■
zum Steuern, Messen und Regeln mit acht S. Lehrheft nur 20 DM, Dipl.-Ing. (FH) Merkl, Innsbruckerstr. 47, 7100 Heilbronn

DIRECTORY-SORT.-PRG. 1000 Files >5 Sek! DM 15. Selbsterkl. TEXT-PRG.+ SORT. Routine >5 Sek! DM 15. Beide DM 20. Scheck/Schein. C 64/Disk.
G. Spaeth, K.bacherw. 9, 8438 Berg

★ Computer defekt? Falscher Resettaster? Der einzige mit Schutzschaltung bei mir 10.- DM für VC 64/20 ★ Christian Wöhler ★ Moritzstraße 70 ★ 4300 Essen 1

C 64: Frei definierbarer Zeichengenerator+1 K **mehr** Basicspeicher+Platz für 16 Sprites! Alle Pokes & Anleitung für 10-DM-Schein: A. Ost/Neuweiherstr.3/ 8500 Nürnberg 30

Wer aus dem Raum München kann mir Tips und Tricks für den C 64 verraten? Bitte schreibt mir!!!
G. Völkening, Kemptener Str. 5, 8000 München 71

Habe Spiel- und Progr.-Beschreibungen und Adventure-Lösungen !! Info gegen Rückporto ! W. Kokott, Lohengrinstr. 11, 8034 Germering

Wer tauscht Turbo-Tape-Steckmodul? Biete Soccer-Steckmodul
Udo Jensen, Klockries 26, 2263 Risum-Lindholm

Verkaufe wegen Systemw. C 64, 1541 Datas., Monitor, Drucker GP 100 VC (alles 3 Mon. alt), Lit.+Orig.prg. — Knüllerpreis DM 1999 — E. Wiedmann jun., Urbanstr. 13, 7132 Illingen

Suche Tauschpartner zwecks Programm-Tausch! Nur Diskette! Kaufe keine Programme! Listen an: Sönke Engwer, Waldenburger Str. 37, 4630 Bochum 1

★Superangebot★★★Superangebot★
Verkauf von 50 Disk mit ca. 750 Spielen wegen Hobbyaufgabe. VHB bei 1000 DM. Auch viele neue Prg. Info bei PLK 051502; 29 Oldenburg

■■■■ Suche **Anleitungen** ■■■■
Wordpro 64, Kontomat, ISM 64, Blitztext, SM-Joker 64, Vizastar 64 u.a. — **Nur Tausch** mit PRG n. Wahl — Andy Wissing, 44 Münster, Spichernstr. 16

HOBBY-AUFGABE: Verk. meine C 64-Prg. 67 Disks für 680 DM+10 Kass. für 150 DM alles 83/84. Verk. auch einzeln Disk/Kass. = 20. Tel. 0421/611414, D. Scheel, 2821 HB, Gröpelinger Heerstr. 369

Trainerpokes für 15 Spiele (Manic Miner, Snokie, Hunchback usw.) und fast alle Adventurelösungen. Liste bei Erik Schörger, Sallstr. 73, 3 Hannover 1, Tel.: 0511/858757

C 64! SIE HABEN FORT APOKALYPSE u. kommen nicht weiter? Ich habe den Poke für 98 Hubschrauber! 10 DM (Schein) an W. Roth, Kulmbacherstr. 3, 8641 Marktrodach

Übrigens, dieser POKE funktioniert nur bei der einteiligen Version von Fort Apokalypse und wurde bereits im 64'er 9/84 auf Seite 96 veröffentlicht. Die Red.

Suche defekte Commodore-Computer/-Peripherie und andere. Melden bei Stefan Luik, Grüner Weg 14, 7410 Reutlingen 17, Tel.: 07121/17805

Suche Pascal-Version für Commodore 64. Angebote an
Winkler Martin
Römerstraße 58
5427 Bad Ems

★★★★ SUCHE !!! ★★★★
Anleitungen und Adventures aller Art. Anleitung als Original/Kopie. Angebote an: M. Drews, Aurikelweg 135, 5024 Pulheim 1

C 64 mit Totalschaden,
Schaltpläne von C 64, VC 20 und von Floppy 1541 gesucht.
Angebote unter: Tel. 07393/812 ab 17 h

Verkaufe C 64
Monitor, Datasette, Joys, Software (Turbo, Simons, Exbasic, Spiele...). Literatur (u. a. alle!!! 64'er) neuwertig, VB 1200 DM, 04192/9512

TAUSCHE C 64-Software-Kass./Disk!!
Suche Flight Simulation II mit Handbuch!!
Angebote mit Liste an: Martin Suchy, Breslauer Str. 28, 4507 Gaste, Tel.: 05405/2732 Eilig!

Dienstreiseabrechnung als Anlage zum Lohnsteuerjahresausgleich für C 64. Disk gegen DM 15.-
Schein/Scheck bei Rolf Heister, Böttgerstr. 29, 5205 St. Augustin 3

★Achtung: Tausche Programme (Disk★
★ und Tape)/Tausche auch gegen ★
★ Bücher!/Über 900 zur Auswahl ★
★ Stephan Cattau, Pastorenkamp 8 ★
★2841 Wagenfeld/Tel. 05444/1811★

Grafik Hardcopy und Original C 64-Zeichensatz für CP 80 über Userport, Software-Interface mit ausführlicher Anleitung, Software+Verbindungskabel DM 80,- — 0237/84614

C 64 C 64 C 64 C 64 C 64 C 64 C 64
Suche Software aller Art! C 64
Schickt Eure Listen an:
★★★ Lars Jedinski ★★★
★ Tilsiterstr. 40★4650 Gelsenk. ★

Bücher-, Platten- und Konto-Dat. je Datei DM 25,- ★ Info ★ gegen Freiumschlag. A. Lübs, 21 HH 90, Kerschensteinerstr. 17 ▶▶ neu ▶ Videodatei DM 30,- ◀◀◀

COMPUTER-MARKT

Private Kleinanzeigen

■■■ Akustikkoppler C 64 ■■■ 300 Baud, Vollduplex inkl. Gehäuse, Software, Garantie! Preis: DM 350,-, anschlußfertig. Info unter: 04531/85755

Suche mech. def. Floppy bis 100 DM oder kostenlose Spiele an Klingmüller, Wehrstr. 7 5330 Königswinter 41 oder 02223/21669

Suche Floppy 1541 bis 350 DM Heiner Kraus Margaretenstr. 2 6689 Merchweiler Tel. 06825/45292

Suche Schach-Turnierprogramm — Schweizer System — bis zu 9 Runden mit 50 Teilnehmern oder mehr Angebote an Rauschenbusch, Tel. 07066/7346

SucheSucheSucheSucheSucheSuche **Teleterm** und andere Treibersoftware für DFÜ mit dem C 64 Angebote an: H. Rattenholl, Bundesstr. 17, 4804 Versmold (?TAUSCH?)

Verk. C 64, Floppy, 1541, 2 Joyst., Disk., Prog., Bücher für kompl. DM 1000.- (5 Mon. alt) Tel. 09636/763 nur Samstags u. Son. Schicker Josef, 8591 Schönkirch

★★★C 64 + 1541 + Datasette★★★ + Joystick + Lightpen + Software (Ex-Basic, S.Basic, Forth, Ass., Mon. Spiele u.a.) + umf. Literatur für 1400 DM zu verk. Tel. 02053/7927

Verkaufe meine gesamte Software! Info gegen 0,80 DM von: Postlagerkarte 051525 C 2900 Oldenburg

Suche und tausche C 64-Software Liste an oder gegen 80 Pf von: CSO Postlagerkarte 099912 A 2300 Kiel 17

Schüler sucht defekte Floppy für C 64 mit behebbarem Schaden bis 300 DM. Harald Schmitzer, Leibnitzstr. 33, 4020 Linz, ÖSTERREICH

THE DALLAS QUEST !!! Verkaufe Lösung für 5 DM. Schein/Scheck an: R. Daniel, Raiffeisenstr. 52, 8753 Obernburg 2. Bin sehr am Tausch von Spielen interessiert.

Suche Software für C64 auf Kass. z.B. Logo, Pascal, Flight II, Sportspiele, HES-Sunner, G. Adventures Ultima I-III usw.! Anleitungen: M. Schürmann, Kleeweg 29, 4973 Vlotho

Suche Top-Software auf ★ C 64 ★ nur Diskette, schickt alles, Liste mit Preisen an: ★★★★★★★★★★ Thomas Wellpott ★★★★★★★★★★ Dr.-Max-Ilgner-Str. 5, 4992 Espelkamp

Commodore 64 + Floppy VC 1541 + Datasette + Monitor + TV + Timer-Fisher + Schreibtisch + 12 Volt Netzteil-Autoradio + Joystick f. kompl. DM 2.100,—. Tel. 040-8604098

Verkaufe Roos-80 Zeichen/Grafik-Karte mit Textverarbeitungsprogr. für C 64 für DM 200.- Tel. 06103/27140

C 64+1541+1526+Zenith ZVH-122-E 1 Jahr + Software + Literatur, komplett VB 2000 DM o. einzeln zu verkaufen. Ab 17 Uhr Telefon (02245) 2420

VC 1530 Data mit 400 Progr. für VB 200 DM. Uwe Dahlhues Suedlohnerweg 11, 4424 Stadtlohn, Tel. 02563/7499. Su. Softw. Listen bitte schr. Biete für 1541 Floppy 500,-

Lernprogramme: z.B. Zeichensetzung m. Kommaregeln, Autorenprogramm (Sie erstellen Ihr Lernprogr ohne Programmieren). Info 50 Pf, G. Wagner, Boddinstr. 60, Berlin 44

Biete: **FESTABA C 64** Univers. Statik u. Festigkeits-Prg., 2-D FEM, Stab, Balken, Pkt.- u. Streckenlast, Dipl.-Ing. H. Krohn, Auwiese 13, 2080 Pinneberg

Wegen Systemaufgabe C 64 und Drucker MPS 801 für komplett 1000 DM zu verkaufen, ca. 6 Mon. alt. Gerd Hessenbruch, 2 Hamburg 71, Brachetweg 7, Tel. 040/6427329

C 64-Software auf Kass. o. Disk. Umlaute öÖäÄüÜß für C 64 .... 20 DM VIC1525 als Schreibmasch. ...... 30 DM Info anfordern bei Schuhmacher, 62 Wiesbaden, Biebricher Allee 81

Verk. neuwertig: VC 64 DM 500+VC 1541 DM 500+Monitor DM 200+Drucker 1526 DM 500+incl. Softw. (Textomat, Datamat, Supergrafik, versch. Spiele +2 Joyst.+Bücher), Tel. 06103/33527 (Dreieich)

C 64 Digital-Uhr C 64 ROM-Karte anschl.fertig! Daueranzeige abschaltbar. Über Basic einstellbar! Freiprogr. Statuszeile definierbar. Basic kompatibel 99 DM 0234/411913

Suche prof. Programme (z.B. Datab), Text (mit dtsch. Zeichen), Pascal, Daten und Tabellenkalkulation (mit Tortengrafik). Simons B., Tel. 0471-38945

Check 64 testet Ihren 64+Zubehör softwaremäßig. Kass. od. Disk. nur 20 DM. Reset zum Stecken 10 DM im voraus: Wolfgang Hewig, Postdamm 18, 4460 Nordhorn: ★DM nur im voraus!

DRUCKE Ihre Listings o. Textseite v. Disk je Seite (A4) nur 15 Pf + Rückumschlg.! Bearbtg. am Eing.-Tg.!! Angebot gilt immer. Garantiere ©. Baade, Forststr. 120, 495 Minden

C 64+Datasette+Joysticks+Spiele z.B. Schach, Fort Apokalypse usw. Simons Basic+Buch und weitere Literatur über Basic allgemein, Preis nur 800 DM VB, Tel. 5330 Mulheim/R/755768

Tausche C 64-Software, ca. 450 Prg. vorhanden, Tel.: 332232, Vorw. 0203, Karl-Heinz Hillus, 41 Duisburg 1, Mülheimer Straße 57

★★★ MÖNCHENGLADBACH ★★★
★★ An alle 64er Freaks. Meldet ★★
★★ Euch. Es lohnt sich. ★★★
★★★ Listen an: W. Koenes ★★★
Hauweg 31, 405 Mönchengladbach 1

★Suche defekte C 64, möglichst keine Tastaturfehler (Tasten fehlen). Anschriften an Royal-Soft, A-3100 St. Poelten, Postfach 613, auch deutsche Zuschriften!!!

Commodore C 64, Datasette, Joystick, Paddle, Spiele, Disketten, Literatur, 900,- DM, Tel. 0711/7909736 und abends 457565, Liebgott

Private Kleinanzeigen

Epson CX21 Akustikkoppler mit Interface f. C 64 v. Orig. Teleterm Commodore Drucker 801
Beides 2 Monate Garantie 1000,- DM
H. Thelen, Tel. 0221/3604729

C 64 Kopierschutz für Ihre Software. Weder block- noch programmorientiertes Kopieren möglich!! Info gegen DM 1,- in Briefmarken. R. Hobmeier, Lothstr. 100, 8 München 40

Suche Commodore 64 mit Totalschad. oder nur das Gehäuse eines C 64
Biete bis 150 DM R. Metzler, Schützenstr. 6, 7734 Brigachtal
Tel.: 07721/4821 ab 18 Uhr

User-Club C 64 Dortmund
sucht noch Mitglieder, möglichst Raum Dortmund. Wenn Interesse, dann Di-Fr 16.00-19.00 Uhr unter DO 593163 anrufen! Matthias Günther

Suche gebr. C 64, evtl. mit • • • • • Datasette • • • • • • • • • • • • • •
Jürgen Brunner • • • • • • • • • • •
Hiltpoltsteiner Str. 23a • • • • • • •
8500 Nbg. 10 • • • • • • • • • • •

VERK. Adv.-Lös. zu Dallas, Gruds, Crit. Mass, Aztec T, Mis.Asteroid je 4 DM. Deutsche Anl. zu Flight Simulator 2 = 15 DM. V. Mitlacher, Sulzenstr. 8, 8624 Ebersdorf ★★★★★★

Verkaufe Orig. Simons Basic Disk + Handbuch für DM 50,- und Kass.-Spiel Skramble DM 10,-
H.M. Dechau, 2000 HH 74
Tel. ab 18 Uhr 040/6513211

Verk. Grandmaster-Anleit.: DM 10, Lösung v. Aztec Tomb: 5 DM+RP.
Wer hat Erfahrung im ROBOTERBAU?
Franz Gaßner
Hauptstraße 40, 8899 Aresing

Suche günstig Compiler-Basic 64 mit Handbuch von Data Becker.
Suche Software, auch Anwender-Prg. (günstig). Angebote an: K. Schmid, 8951 Aufkirch, Helmishofenerstr. 17

★★★ EPROM-BÄCKEREI ★★★
Wir backen Eure Prg. auf EPROM!!! Kein Programmverlust mehr. EPROMs und Platinen billig!!!
Tel.: 0208/621 64 + 0203/773042

SOFTWARE

Suche Software aller Art für **C64** auf Kass. oder Diskette. (Besonders Sprachu. User Software). Liste bitte an: Dörflinger V. c/o Pichler, A-9844 Heiligenblut / Österr.

■■■■ Zuviele Programme? ■■■■
CS-Sort., das Prg. zur Verw. Ihrer Prg's. Suchen, Selekt. Speichern. Preis: VHS! Original! Ch. Stahlberger, Pfädlistr. 4, 7858 Weil. # 3. $

Suche Anleitungen (Kopie) zu Archon Mule Flight 2 Wordpro J. Mission Benji Synthymat Musik C-S. usw. Bezahle gut! C. Lagneau, Elektrastr. 11, 8000 München 81

Achtung! Ich habe für Superbase nur eine engl. Anleitung. Wer kann mir gegen Kostenerstattung eine deutsche überlassen? E. Schambeck, Emskirchnerstr. 4, 8531 Hagenbüchach

Suche Software für C64. Angebote und Listen an Hartmut Brinkmann, An der Heideschule 3, 4508 Bohmte 1

■ Massig Infos u. Paperware C 64 ■
Hilfe, Platzmangel. Fachlektüre z.B. Chip, HC, Elo, elektor uva. (83/84) billig. Liste gegen adr. Freiumschl. J. Renfer, Ebertstr. 43, 67 Lu-hafen

Super Lern- und Spielprogramme für den C64 zu verkaufen. Info bei: Uwe Maurer, Steinheimerstr. 9, 6229 Oberwalluf, Tel.: 06123/72525

Verk. Original Simons Basic Disk mit deutschem Handbuch DM 100,—. Centronics Softwareinterface für Epson Drucker. A. Reichensperger, Dettenharter Str. 12, 8851 Tapfheim

Achtung Österreicher
Verkaufe und tausche Supersoftware für C64. Eduard Müllner, Rudolf-Zeilerg. 67/5/1, 1230 Wien/Österreich

★Anfänger★ ★C64★
Top Software gesucht ★ Suche Tauschpartner für Software ★ Tel. 0461/26812 ★ G. Maske ★ Kanzleistr. 27 ★ 239 Flensburg

Suche Prgr. zur Durchführung eines Schafkopfturniers mit Punkte, Zählung/Teilnehmerlisten/Siegerermittlung. Angebote an: M. Steinbach, Fastlinger Ring 167, 8044 Unterschleißheim

Suche gute Hardcopy für Plotter Neumüller Npr. 5500 (Adcomp X100) mit Interface Typ 9200/4 für C64. J. Woitschätzke, Tatzelwurmweg 7, 8022 Grünwald

Suche die neuste Software für C64 auf Kassette z.B. Summer Games u. Adventures. Verkaufe ZX81 und CBS-Spiele. Liste an: K.-H. Jahns, 4 Düsseldorf, Burghofstr. 58

An alle C64-Freaks
Suche Top Games/Adventures für meinen C64 (Tape). Listen an J. Pieper, Schleusenstr. 32, 2178 Otterndorf

Mein C64 hat Geburtstag. Suche neue Software um ihn damit zu füttern! Nur Tape! Sendet eure Listen an Helmut Schmitt, Erlkastr. 33, 2110 Buchholz, Tel.: 04181/5966

C64 — Suche Top-Programme (Grafik, Adventures, Musik) nur Kass.! Schickt Listen an Frederico Helfrich, Zentnerstr. 9, 8000 München 40

C64 ★Pro-Vok Ultiomo★ C64
Menügesteuerter Vokabeltrainer, Editor + Trainer + Druckerfunktionen. B. Stiehl / Str. d. Rep. 49 / 6200 Wiesbaden Tel. 06121/66028 — Prg. + Disk = DM 25,— + Porto

Test- und Demodisk für VC 64 mit 50 Programme für DM 30,— + NN. B. Kock, 4592 Lindern

Mache Sicherheitskopien von ihren Disketten und schütze ihre Progr. gegen Löschen! W. Kokott, Lohengrinstr. 11, 8034 Germering

Private Kleinanzeigen

Suche f. C64 + 1541 Software für: Statik, Architektur, LV's, Ekst., Buchführung, Text + Adressen, Sprachen, Vok.; Angeb. an P. Junge, Wühlestr. 26, 7320 Göppingen

★ Neue und gute Software für C64 ★ wird gesucht. Disk und Kass. Eure Listen, auch wenn ihr tauschen wollt, an: Michael Marek, Ulmenstr. 2, 8031 Eichenau (08141/87 71)

Lösungen für viele bekannte Grafik Adventure z.B. Ulysses, Grads in Space, Hobbit und Dalles. Wolfgang Pützold, Haffweg 20, 3004 Isernhagen (pro St. 5,— DM)

★Achtung C64-Besitzer, vergessen Sie die doofe SCHRIFT-Wiedergabe, schärfer geht es nicht mehr mit neuem Zeichensatz — Epr. U5, DM 60, auch m. Uml. Tel. 09721/62346

★★★THE HOBBIT bezwungen★★★ Wer hängt bei HOBBIT? Ich habe die Lösung!! (auch f. and. Adv.) Suche außerdem Time Zone u.ä. f. 64. Tel. 02243/2112 tägl. ab 13.00 Uhr

Vizastar u. Superbase 64 einschl. Anleitung gesucht. Kauf o. Tausch. Datamat neu DM 60,— abzugeben. Hans Breuer, Tel. 02162/15864

Block-Graphik-Zeichenprogramm (nur auf Diskette) für Commodore 64 zu verkaufen DM 25,— Oliver Lessner, In der Wann 44, 3550 Marburg, Tel.: 06421/33340

Superpreis: Fasttape, Simons-Basic mit Demo und Handbuch, Grandmaster, 80-Zeichen + 5 Topspiele nur DM 50,— Tel. 0621/708195 ab 17 Uhr oder Wochenende

MPS-802: HARDCOPY-PROGRAMM für Disk, 100 % Mcode, 1- 2- 4facher Hires-Ausdruck aller Bilder, Super-Editiermodus, 20 DM an: E. Stolle, W.B. Feldstr. 51, 7082 Oberkochen

Probleme mit dem 1526? Kein Zeilenvorschub bei umdefinierten Zeichen? Zeichensätze: dt. oder Schreibschrift. Info bei M. Böhmel, Am Töbele 2, 7923 Königsbronn

CAD-System für VC20 mit Selbstbauplotter: Pgm. mit Bauanleitung 50 DM: Weitere Anwender Pgm: Text, Adress u.a.: Info 1 DM in B-Marken. Heinisch, Diemelweg 9, 427 Dorsten 19

Neue Software!! Neue Software!! Diskettenverwaltung bis 3000 Prg. ★Treibersoft für Akustikkoppler★ sowie andere gute Soft ★ Info bei: A. Krause, Brüderweg 60, 59 Siegen 1

Suche Anleitungen für Textprg.: Wordpro3t, Vizawrite, Paperclip64C, SM-Text/Adreva64. Angebote an H.-J. Baumann, 4352 Herten, Telefon: 02366/83031

Suche für C64/1541 Finanzgenie u. Superbase 64 mit Handbuch von Data Becker. Kaufe oder tausche. Angebote an: Joh. Heinen, Grootmoor 174A, 2000 Hamburg 71

Textverarbeitung mit C64 und Schreibmaschinen. Info's gegen Freiumschlag. Thomas Erpel, Berliner Str. 14, 4830 Gütersloh

————Inventur C64———— Machen Sie eine profimäßige Inventur-Auswertung m. bis zu 150 Warengruppen, 10 Lagerorten, 4 Altersstufen u. Teilwertabschreibg. Info.: 04503/3235

Kaufe alle preisgünstigen 64'er Programme. Angebote an: P. Wegscheider, Hochstr. 29, I-39049 Sterzing

Suche SW (Spiele!) für C64 und Atari 600XL. Liste bitte an: J. Gagesch, Neuschwansteinstr. 16a, 8900 Augsburg

★ **C64: Mammut**-Diskverwaltung!! ★ Optimale **Pflege**, große **Benutzerfreundlichkeit**, dazu ein **Superstarkes Copy!** Info auf (+ 80 Pf) bei C. Usaat, Erndtebrücker W. 3, 1 Bln 27

★★★Deutsches Text-Adventure!! Über 150 Wörter! Wortliste + Lösung! C64/VC20 & 28k (Typ angeb.) >Markenkass. = 20 DM (Schein/Scheck): R. Zühlsdorf, Dolfenstr. 22, 4500 Osnabrück

Suche Software für C64. Angebote an: Daniel Vdovic, Mellerstr. 64, 4800 Bielefeld 1

■ **VEREINSVERWALTUNG — VC64** ■ Sort. Mitgliedslisten, Bankeinzug, Rechnung, Aufkleber; nur Diskette DM 69, Info gegen Freiumschlag: H. Schäffer, Bergf. 21, 8261 Tittmoning

Deutsche Tastatur für C64 **äÄöÖüÜß** auf Drucker **(MPS 801/1525/GP 100 VC)** und Bildschirm. Disk, Kass.: 14 DM ★**24-Std.-Schnellversand**★ S. Heupt, Im Grohfeld 25, 6090 Rüsselsheim.

★★★★★★★★★★★★★★★★★★★ SM-Text + SM-Adreva Originalprogramme mit Handbuch! Nur komplett für 199,— DM Tel.: 0681/51294 ★★★★★★★★★★★★★★★★★★★

Verkaufe oder tausche. Original Datamat m. Handbuch v. D. Becker. SM-Joker/64 m. Handbuch (Origin.) Joh. Heinen, Grootmoor 174A, 2000 Hamburg 71, Tel.: 040/6404988

Verkaufe MATHEMAT, Original DATA Becker Programm, zusammen mit Handbuch 60,—. J. Dahm, Untere Bergerheide 1, 5600 Wuppertal-1, Telefon 0202/723334

Imbißkasse, Autostart, VC 64
Diskettenprüfung, VC 64
Lohn- Einkommensteuer 1984, VC 64
DFUE-Modemprogramm VC 64
★★★★★02222/5678★★★★★

★★★C64/VC1541★★★ 10 Prg's die Sie für die 1541 brauchen! Z.B. Copy, Monit., Disk-Datei. DM 45,—★★★ Info gegen Freiumschlag. C.R. Sänger, Nebenstr. 11, 6451 Großkotzenburg

Suche Programm um auf C64 in Pascal arbeiten zu können, sowie B.Liga (Kass.) Info an F. Schmiedel, Valdersweg 46a, 5161 Merzenich Tel.: 02421/36212

C64 — **1541 — User** — C64 **Autostart 1.1 + Disk-Utility + Prg. Verwalt.**! 3 Super Prg. a. Disk m. Anl. **Nur 30,— DM**(Schein/Scheck): Jürgen Rumpel, 87 Würzburg, Herzogenstr. 2

★★★★Suche gute Software★★★★ ★★★★und Anleitungen zu★★★★ ★★★★Spielen und Userprg★★★★ Bitte keine Fantasiepreise. W. Kömes, Hauweg 31, 4050 Mönchengladbach 1

Disk Copy Set: 3 min Copy + Diskmon + RE-Format + Inh. S + Einzel Copy (Prg. Seq+) + Floppy-Mon auf Disk für 35,—. Vork. oder (NN) bei T. Schnebbe, Stralsunder Str. 26, 28 Bremen 21

Private Kleinanzeigen

VC 64
Suche zuverlässige Tauschpartner für C64-Programme auf Kass. Tauschliste an: Christoph Brehm, Hauptstr. 41, 6694 Urexweiler

Wenn Sie kein Geld zum Kauf v. Software, Zubehör und Bücher verschenken wollen, dann .... **Sofort Gratisinfo anfordern von** G. Henning, Pf. 126665, 1 Berlin 12

Die persönliche Grafik für den 64er. Erstelle HRG nach Ihrer Vorlage (Foto, Zeichnung, Dia, usw.) Info Manfred Wyrwas, Dimhausen 56, 2830 Bassum 1, Tel.: 04241-1093

Suche gute C64 Software! Arcade- und Adventures-Games bev. Angebote an: Harald Schleicher, Neherstr. 25, 7700 Singen/Htwl

Akustikkoppler 208,— DM!!! Terminalprogramm für Modem und Akustikkoppler 30,— DM, komplett 228 DM. Info von A. Stocker, Berghausen 13, 5778 Meschede, Tel. 0291/1221

Verkaufe original Textomat, Diskomat, Profimat incl. Handbücher für je DM 45,— R. Bärwolf, Postfach 100321, 4650 Gelsenkirchen.

★ ★ ★ **Deutscher Zeichensatz** ★ ★ ★
(Ä Ö Ü ä ö ü ß) für C64 läuft auf jedem Drucker. Volle Speicherkapazität. 0741/43300 oder 0741/6076

Suche Software für VC64 auf Diskette — spez.: Bruce Lee, 3-D Deep Space, House of Usher. Liste bitte an: Thomas Unger, Uhlandstr. 6, 7434 Riederich

Achtung! Suche Adventures + Spiele mit Anleitung. Angebote an Carsten Treuner, Am Gehrlskamp 2, 3000 Hannover 21, Ps.: Nur Disk.

Su. z. Tausch: Beamrider, Hobbit, Aztec Chall, Bruce Lee, Decathlon Twin, Kingd. Valley, House of Usher, River Raid, Pitfall 2, Space Pilot. An H. Draxler, Falkenstr. 25, 8000 München 90

Börsengewinne! C64 ermittelt zuverlässige Kauf- u. Verkaufsignale. TIMING-Indikator m. graf. Ausw. Info g. fr. Rückumschl. M. Brauser, Sonnenstr. 14a, 2900 Oldenburg

★ ★ **Chemie-Programme C64** ★ ★
Suche gute, brauchbare Schulchemie-Pgr. Info mit Preisvorstellung an R. Bentzinger, Hans-Thoma-Str. 5, 7858 Weil am Rhein

★ ★ Printer-Plotter 1520 ★ ★
Warum selbst Programme für den Plotter schreiben? Ich biete meine zum Tausch an. Wolfgang Schwanke, Erfurter Str. 6, 403 Ratingen

Suche vertrauenswürdigen Tauschpartner im Raum **—Amberg—Vilseck—Auerbach/Opf.**
Liste oder Anruf an: M. Müller, Bergstr. 6, 8572 Auerbach, Tel.: 09643/3147

■■■■ Vokabelprogramm ■■■■
Schüler ■ Lehrer ■; **Languagefriend** ist das Topprogramm ■ Kassette nur DM 10 ■ Info DM 1 ■ Gert Walther, Plettenbergstr. 16B, 2050 Hamburg 80

Multiplan in Deutsch m. Hdb.	**279**
Vizastar (DB + SS + Grafik)	**339**
Vizawrite (ä, ö, ü, ß) Cartr.	**349**
Flightsim. II Sublogic	**168**

Books & Software (09571) 3182

Suche kaufm. Software für C64 mit Disk (Fibu/Lohn etc) mit Beschreib., evtl. auch Sprachen (Logo, Pascal usw.) Telefon ab 18.00 Uhr 02166/602682 (Z80Karte)

Mathematikprogramme für den C64 Integrale, Vektoren, Matrizen, komplexe Zahlen, Kurvendiskussion.... Info gegen Porto bei WKS, W. Kleuser, Am Buernacker 8, 4630 Bochum 6

■■ Suche **Disk-PRG + Anleitg.** ■■
Copymaster, Supermanager, Text C64, Vizastar 64, Lohn- u. Est. C64, Joker 64, Adreßverw. 64; — **Nur Tausch** — — A. Wissing, 44 Münster, Spichernstr. 16

Wer kann mir helfen? Suche Software für meinen 64'er. Kaufe und tausche auch. Tak-Wah Chan, Langemarckstr. 155, 2800 Bremen 1, Keine Anrufe bitte!! Es eilt!!

Der Commodore-Club. Noch Comalo.14 zu haben 10,— DM. Neu Progr. Super-Disk-Verw. 20,— DM auf Disk. Bestl. bei Commodore-Club — — — C.C Aachen, Mittelstr. 13, Tel.: 0241/13947

C64: Graphikprg. **CAD 64** m. dtsch. Anleitung u. Demoprg. ★ 100 % MC ★ 30 DM ★ ★ ★ Keine Raubkopien ★ ★ ★ 30 DM. Auch weitere Y.E.S.-Prg. erhältl. Guido Jung, Am Anger 2, 8103 O'gau

BAUFINANZIERUNG
Effektivzins ★ ★ Tilgungspläne ★ ★ Ect. auch nach Maß: Sabrowsky, Christophstr. 14, 5024 Pulheim 4

■■ Die Softwarelösung f. C64 ■■
Scroll, Print, Cursorsteuerung usw. m. Farbe in Hiresgrafik Prg. auf Kass. 10,— DM-Schein an Grosse, Wilhelmistr. 12, 6960 Osterburken

C64 Tausch 05422/8331
Suche Tauschpartner od. Club. Auf Disk oder Kass. T.-Verhältn. egal. Liste an/von: Horstmann, Wulberg 29, 452 Melle. Antwort garantiert.

C64 6mal Original Data-Becker C64
1. Datamat, 2. Faktumat, 3. Supergraphik, 4. Kontomat, 5. Diskomat, 6. Uni-Tab, a. mit Handbuch. 1,2,3,4,5 je 60,— DM. 6=40,— DM. Tel.: 02369/8862 ab 19 Uhr

Suche für C64 Schreibmaschinenkurs sowie andere Programme auf Kassette. Angebote an Hans-Rudi Hirsch, 7840 Müllheim 15, Höllbergstr. 17, Tel.: 07631/4519

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
Hallo 64er Freak
Suche gute Spiele (nur auf Kassette). Bitte an Dirk Rohde, Burghasungerstr. 11, 3501 Habichtswald od. Wolfager Str. 12b

C64-Kurvendiskussion / S.-Basic s. Komfort, mit Graphik. Kassette o. Disk., Preis DM 30,— (bar oder Scheck) Info kostenlos: H.-J. Diestel, Dorfstr. 79, 2263 Risum-Lindholm

Suche Dimension X, Mask of Sun, Bruce Lee, Summer Games, Lod Runner, Pharaoh's Curse, Archon u. andere Prog. auf Disk. H. Mayer, Wiesenstr. 7, 6719 Ebertsheim 2

Das Optimum!!!
100, 200, 300 Games/Utilities für 100, 200, 300 DM inclusive Disks! Tel.: 07661/5539 ab 16.00 Uhr — 19.00 Uhr. Sofort anrufen!!

Suche für Weinbaubetriebe geeignete Software auf Disks für C 64 (Kellerbuch, Weinverkauf u.a.) Herbert Mauss, A 3493 Hadersdorf, Hamerlingstr. 3, Österreich

Suche Spielprogr. ab + 3K. Bitte Liste mit Preis oder evtl. Tausch an: Klaus Hampshire, Anker Str. 24, 5102 Würselen, oder Telefon bitte bei Auskunft erfragen! Neu

COMPUTER-MARKT

Private Kleinanzeigen

Superpreis: Simons-Basic mit Demo und Handbuch, Grandmaster, 80-Zeichen-Karte + 5 Topspiele nur DM 50,—. Tel.: 06 21/70 81 95 ab 17 Uhr oder Wochenende

Drucke Ihre C64 Prg. von Disk auf DIN A4 für 0,50 DM pro Blatt + 1 DM Porto / Tausche C64 Software. Liste an: Christof Kreutzburg, Mühlenstr. 13, 3055 Hagenburg!!

VC 20 Prg.: Adressenkartei	25,— DM
Gold	25,— DM
Mau-Mau	20,— DM
Verkauf: Kass./Disk., Info	0,80 DM

M. Rick, Denglerstr. 26, 53 Bonn 2

?????C64/Disk?????
Suche günstige Spiel-, Hilfs- und Anwenderprg. Angebote an: P. Vogl, c/o Müller Irmgard, D-8205 Kiefersfelden, Kaiserstr. 14

C64 **AMWAY-Geschäftsprogramm** mit automatischer Bestellung, Bestellausdruck / 14 + 7 % Mwst / Beraterkonten / Fehlbestände / Bonusabrechnung / Lagerverwaltung Tel.: 04 51/39 44 70

Wegen Systemwechsel Original SM + Data-Becker Software + Bücher, Flugsimulator, Decathlon, Grandmaster usw. Tel.: 089/40 55 66 od. Liste von Etzler, Josef-Mohr-Weg 50, 8000 München 83

Anfänger sucht Programme aller Art auf Kassette C64 + Info über schnelleres Load, Save + Resettaste. Suche günstige Joysticks. Liste an F. Grivec, 413 Moers 2, Bendmannstr. 36a

Achtung C 64 Fans: Verkauf gegen Höchstgebot alle Ausgaben 64er Magazin, Software: Superbase 64, Datamat, Faktumat, Kontomat neue Vers. Tel.: 06221/20209

Suche Textverarbeitungsprogramm mit Anleitung (D) das auf Epson RX80 mit Wiko-Softw.-Interf. lauffährig ist. F. Kannebecker, Hardtstr. 27, 5160 Düren 17

Verkaufe Prg. für CBM 64! Alle Prg. sind von mir erstellt (z.B. Koala-Show I + II, Black Oil, usw.) Info gibt es bei Ralf Höhne, Friedhofstr. 34, 4223 Voerde 1

Suche das Spiel Summer Games + Turbo Tape für den C64 + Datasette. Wer verkauft oder tauscht? An Christian Elsner, Steinbruchstr. 2, 4811 Oerlinghausen. Tel.: 05202/1257

Master-Tool, das unentbehrliche Werkzeug für jeden CBM 64-User jetzt nur 29,80 DM! Trace, Renumber, Floppy-Bef. usw. Mono-Soft. Stockmannstr. 29, 8 Mü. 71

Verkaufe mein DATA-BECKER Orig.-Programm Profimat + Handbuch (Orig.-verpackt) für nur 58,— DM
Tel.: 089/797709

C64 Deutsche Opt.-Scheine. Info gg. DM 3,— in Briefmarken. Programm auf Disk. DM 40,—; Kassette DM 35,—.
H. Dauth, 6663 Dellfeld, Bergstr. 27

Kaufe Software aus den Bereichen, Wasserbau, Tiefbaustatik, Baust. f. C64. Angebote mit Progr.-Beschr. an ★ Siegfried Bagdahn, Am Park 5, 3341 Groß Denkte ★ Tel.: 05331/69437

★ ★ ★ An alle Hobbyelektroniker ★ ★ ★
Kohle-Metallwiderstände / Dünnfilm-Kondensatoren Bestimm Progr. Farbcode + Zahlencode Eingabe 7,— Lars Hauck, Am Weigert 8, 8721 Zell

Suche Floppy und Software für Commodore 64. Angebote und Listen an Rüdiger Urban, 4000 Düsseldorf, Marconistr. 9. Tausch und Verkauf von Software billig und sehr gut!

Suche Chemie-Programme für C64 evtl. Tausch. Angebote an Thomas Kraul, Otto Suhr Str. 22 2/106, Tel.: 04 41/59 12 29

Wären Sie gerne Millionär?
Wir bieten Ihnen alle Lottoziehungen 6 aus 49 ab 1970. Sequ. Datei auf Disk. (1541) + Hardc. incl. Porto DM 30. Tel.: 07457-1245

Suche Business-Progr. für C 64: Buchhaltung, Fakturieren usw. F. Jaschusch, 48 Bielefeld 18, Alte Detmolder Str. 2

Verk. Summ. Games, S-Basic, H. Olymp, Dal Quest, Flight II, Bruce Lee, Grandm., Jungle H., Paint M., Zaxxon auf Disk! Je 2 Pr. 50 DM Vorkasse. Schmitt, Krayerstr. 14, 4300 Essen 14

Telekommunikationssoftware z. Kommunikation m. Mailboxen ★ 100 % MC ★ 60KRam f. Daten ★ Funktionen: Text vorschr. ★ Speichern u.a. ★ DM 33 + NN ★ M. Güttner, Wengerfeld 4, 8116 Eschenlohe

Suche Software (keine Spiele) Techn. Berechnungen, Klima + Heizung, Bauwesen etc. Tel.: 07223/26734

Achtung! Achtung! C64-Besitzer!
Suche dringend auf K. Spiele u. Adventure wie Dallas, Summer Games usw. Bitte anrufen: ab 18 Uhr. Tel.: 05374/2309

Kappes Data präsentiert das Weihnachtsgeschenk zum C64 zum Sonderpreis von DM 16,— inkl. Disk. **Datei 64**, das Dateiverwaltungsprg. Stegstr. 77, 6000 Frankfurt 70 ★★★★

Die Mailbox für den C64 zu verkaufen mit Koppler Epson CX-21, mit Orig. Teleterm und RS 232 mit Automodem und Wählautomat mit Mailboxsoft sofort Betriebsbereit und auf alles noch 4 Mon. Garantie >>>Preis VHB<<<
Tel.: 040/731134 88 od. in einer der HH-Boxen für CATWIESEL

Neu!!! Riesenbuchstaben drucken, Schriftsätze erzeugen m. C64/RX 80. Versch. Schriften (Block, Gotisch, u.a.) Info DM 1,40 von D. Schulze, Rebenring 631027, 33 Braunschweig

64er Softwaretausch (Disk) Topspiele, keine finanziellen Interessen. Liste an Uwe Riese, Auf der Linnert 10, 46 Dortmund 76, Bitte keine Telefonanrufe

Akkordabrechnung! Mit Kostenstell. für Gruppenakk. u. Kontr. d. Ges.-Std. Ausgabe auch a. Drucker. Info gegen Rückp. DM 1,— D. Ganderke, Marschacht 1, Amselweg 1

★ ★ ★ ★ ★ 64'er Software ★ ★ ★ ★ ★
Suche Software auf Disk. Bruce Lee, Dimension X und gute Sportspiele. Melden bei Tillmann Klepsch, Tel.: 04 51/50 18 22 (abends)

Suche Anleitung Gruds in Space, Spiel Beach Head, Programm zum ID ändern incl. Direktori. Anleitung zum Diskdock in deutsch. Klaus Appeldorn, 2830 Beckeln. Tel.: 04244/7487

Simons Basic
Originaldiskette und Handbuch für DM 90,— bei Alban Huber, Frickenwäsele 34, 7997 Immenstaad, Tel.: 07545/6421

● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
C64
Suche Tauschpartner für Software (nur Kassette) Niko Kailuweit, Brandenburgerstr. 73, 2110 Buchholz

Suche Multiplan und Vizastar für Commodore 64 — Angebote an Tel.: 06881-1273

Verkaufe Original Simon's Basic, Diskette mit Handbuch (3 Monate alt, kaum gebraucht), öS 1000,—. Zuser Peter, A-8950 Stainach 346,
Österreich Östereich

Anfänger sucht Programme für den Commodore 64. Leider ist kein Tausch möglich. Ich suche auch nicht nur Spielprogramme. Novak Peter, Wiesenweg 3, 8481 Eslarn

Suche gute Adventure- und Fantasyspiele f. C64. Nur a. Kassette! Angebote bitte an: Rainer Paals, Stormstr. 7, 4573 Löningen

FIBU C64
Grundbuch bis 120 Konten, G + V, Jahresabschluß, Summen-, Salden-, Schluß-, Erfolgsbilanz. Drucker: 1526 od. MPS 802. Tel.: Köln 723000

EPSON-C64 Softwareinterface über Userport, Hardcopy, CMB-Schrift, Listings., ausf. Anl. DM 69,—. Info gratis: B. Natter, Wallbergstr. 14, 8011 Höhenkirchen

Suche Tauschpartner im Raum Bonn-Siegburg (Datasette) Tel.: 02244/3995 Michael — ab 18 Uhr

Achtung Fort Apokalypse-Besitzer. Verkaufe für 10 DM den Poke für 98 Hubschr. Suche Softw. für C64 nur Kass. Kay Giesen, Reifferscheider Str. 10, 5372 Schleiden

C64 Arithmos 99 Complex Numeric Process. System, math. Superrechner, 99 Funkt., 32 KB! DM 59 + 10 Disk. + Vers.-Exclusiv durch C. Schröder, Rablstr. 43, 8000 München 80 (Info 1 DM)

Private Kleinanzeigen

Verkaufe Modelleisenbahn **HO (Fleischmann)** sowie Carrera-Servo. Ruft doch einfach an. Bernd Gugel, Tel.: 09104/2927

Urheberrechtlich verfolgte 64'er Fans wenden sich an Walter Wehrenberg, 2808 Syke, Bahnhofstr. 16, F. 04242-50629

■■■■ C64 Magister C6 ■■■■ Bestimmung lateinischer Verbformen in Sekundenschnelle. Info gegen Freiumschlag bei Stefan Koos, Gatower Str. 104, 1000 Berlin 20

★ ★ Der neue Maßstab: ★ ★ ★ ★ VIC-20 Fast-Grafik-Support ★ ★ z.B. Kreise 30 ★ (!) schneller als 64 mit Sim. Nur f. VC + 3 + 8K min. N. Tesch, Friedlandstr. 76, 53 Bonn 1

Adventure-Spieler werden Ihre Freude haben, Regieren Sie Ihr Volk (1b. 6 Sp.) versuchen Sie (C64) König zu werden. 10 DM für Kass. M. Schmidt, Bornsc. 16, 5204 Lohmarz 1

Suche gute Anwendersoftware für C64. Liste an Michael Gschwender, Dorfstr. 3, 8968 Durach

Suche Programm zur Dekodierung von Morse- und Rtty-Signalen für C64. W. Brächter, Marktplatz 5, 6900 Heidelberg, T.: 06221/15975 od. 14567

Gesucht: Progr. für VC 20 oder 64 Skat-spiel
Angebote mit Preis an Heinz Baumann, Hauptstr. 92, 6454 Bruchköbel. Tel.: 06181/76573

Verkaufe Originale Profimat, Galaxy, Encounter, Arcadia; Module: Soccer, Avenger. 41, neue Basic-Bef. mit MS-Basic! Ab 14 Uhr ★0761/276059★

Suche Datamat und Kontomat von Data Becker mit Handbuch usw. Uwe Grabowski, Bösingfelderstr. 7, 465 Gelsenkirchen. Tel.: 0209/370771

Verkaufe Nährwertberechnungsprogramm für den C64 mit Floppy. Jürgen Stocker, Am Hardt 33, 7777 Salem 1, Tel.: 07553/8729 nach 19.00 Uhr

MAT: C64-Prg. zur Matrizenrechnung: UPN-Rechner (Terme m. Matr., Vekt., Skal.auswerten), linGls, Inverse, Determ. u.a. Disk + Dokum. 20, Info 1 DM. H. Strauß, Petersilienstr. 1-3, 33 BS

Verkaufe gegen Gebot (ab DM 50,—) Modul für SIMONS BASIC, SIMONS-Compiler und für Exbasic II für CBM 64: Meinecke, 5000 Köln 60, Phil.-Wirtgen-Str. 26

★★★★★★★★★★★★★★★★★
Gelegenheit
Tausche große Softwaresammlung gegen guterhaltenes Floppy 1541
Tel.: 06131/58169

Verk. Orig. Sim. Basic 100 DM
Verk. Orig. Profimat 50 DM
Spiel Kong 64 10 DM
Alles Original, keine Kopie
Tel.: 069/618263

Ich gebe es auf! (Seuftz) Verkaufe meine 12 vollen Disketten für nur 300 DM (mindestens dreifacher Wert und alles nur Top Programme aus den USA) Tel.: 0511/668754

Suche ständig die neuesten Topprogramme auf Tape für C64. Zahle bis zu 6 DM für ein Super-Programm: Liste an Thomas Suhm, Marktplatz 16, 7614 Gengenbach

VC 20

Eilt! Suche Farbmonitor sowie Floppy 1541 (für VC 20), Tel. 0641/42785

COMPUTER-MARKT

Private Kleinanzeigen

Verkaufe VC 20 + 16 KByte Erw. + über 100 Progr. (Scramble, Tron, Thor, usw.) Preis 380 DM ab 17 Uhr, Tel. 08505/561

Verkaufe VC 20 + Datasette + Erweiterungsplatine + Quick Shot 2 + Listings + Progr. VB: 400/Verkaufe Atari 2600 + 9 Kassetten ohne Joystick. Tel. 0201/578055 oder 02324/24817 ab 14 Uhr

★★ Achtung VC 20-Supermodul ★★ 3 K + Grafik + Monitor in einem Modul, ROM-Karte für 2 ROM, Commodore-Schachcomputer zu verkaufen, Freitag ab 14.00 Uhr, 07721/62842

Gesucht wird! Die deutsche Spielanleitung für The Colonels Houses (VC 20) geg. Belohn. Wer kann helfen? Holger Bautz, Oejendorferweg 58, 2000 Hamburg 74, Tel. 040/7338378 ★ VC 20 ★ VC 20 ★

Österreich: Eine Wucht! Verkaufe supererhaltenen VC 20 + Datasette + 2 Module + 2 Spiele + Basic-Kurs + viele Bücher, Eder Christian, NÖ St. Pölten, (02742) 646505, LBSDBST 17 B

Commodore VC 20 plus Zubehör gesucht. Tel. 0651/16366

VC 20 = 3 K Supererw. + 16 K + Software + Literatur in Originalverpackung NP 800,— VB 480,—, Telefon 0561/496880

Kein Basic mehr, sondern
★★★ Forth ★★★
Audiogenic-Forth-Modul + Eng./Dt. Handbuch + Forth-Buch für 100,—. Tel. 06261/62221

Verkaufe Modulbox VC 1020 DM 150, Rainer Großmann, Haus-Berge-Str. 145, 4300 Essen 11, Tel. 0201/660130

★ Achtung ★
Suche dringend Speichererweiterung für VC 20 (32 K/16 K) gebr. zu kaufen. S. Fox, St.-Beissel-Str. 2, 4232 Xanten, Tel. (02801) 5114

VC 20 + 32 K + 8 K + Datasette + Reset-T. im Tragekoffer + ca. 450 Prg. + Floppy + Literatur = VB 1450 DM. M. Rustemeyer, 3550 Marburg, Ernst-Reuter-Str. 9/oder: 0511/5325582

Verk. VC 20 (6 Mon.) + 27/32 K + 3 K + 210 Programme, darunter Schach, Scrambles, 2 Frogger, Basic-Kurs, Masch.-Sp.-Mon., 3D-Irrgarten usw. VB 540 DM, Tel. (0791) 54195 Schw. Hall

VC 20 + 3 K + 16 KR 450,— DM, Joys. 25,— DM, Floppy 650 DM, Datas. 90,— DM, 3 Kassetten orig. 40 DM, 1 Buch Data Becker 40 DM, alles neuwertig, Telefon 0228/443800

VC 20 + Datas. + Joys. + Basic-Kurs + 70 Sp. + Modul + 3 Bücher, 4 Monate neu: 734 DM, heute: 525 DM, f. **250 DM**
★★ 07254/8607 ★★ EILT!

Suche VC 20
Datas.-Sp. Erw.-Modulbox-Progr. — Module, Epromer, Literatur, etc. W. Sänger, 8510 Fürth-Poppenreuth, Kronst. Str. 10 d, Tel. 0911/795446

VC 20 VC 20 VC 20
VC 20 + Literatur + 1 Modul + viel Software (Basic-K., Scramble, Amok, Pacman, usw.) = 250 DM VB. Melden unter 02241/66254



Verk. VC 20 + 8 KByte + Data-Becker-Buch neuwertig in Originalverpackg. VB 400 DM (NP 500 DM) kein Festpreis! — mit vielen unterschiedl. Progr. Tel. 02373/71398 (als Geschenk!)

VC 20 Suche Speichererweiterungen 3 K bis 32 K, Schaltpläne, Layout, Grafik bis 40 Zeichen, Schnittstellen, Poke's Literatur + Erfahrungsa. Jo. Förster, Reichweinstr. 5, 6951 Neunkirchen

Loderunner und **Shamus**
255 MEN (+ Level)
Die irren **Pokes** für den **VC 20,** 5 DM Schein/Briefm. an A. Gruner, Quellgrund 17 B, 2104 Hamburg 92

★★ Der neue Maßstab: ★★
★★ VIC-20 Fast-Grafik-Support ★★
z. B. Kreise 30mal (!) schneller als 64 mit Sim. Nur f. VC + 3 + 8 K min. N. Tesch, Friedlandstr. 76, 53 Bonn 1

★★★★★ **VC 20 32 K** ★★★★★
+ Datasette + Software + Bücher zu verkaufen für DM 400,—, Peter Rauscher, Linderweg 74, 5000 Köln 90, Tel. 02203/61417

Einfache Synth.-Software-Stimme Pgm. 1: Lauteing. m. D'sette, Wiederg. per TV. P2:30 gespeicherte Laute. 10 DM/Pgm.Tausch. Info geg. R-Porto. J. Schelbe, Prosper 3, 8058 Erding

Verk. VC 20 + 16 K + Masch. Spr. Modul + Handbücher + Tips/Tricks + Software (auf Kass.)
VB: 350,— (Neupreis: 650,—)
Tel. 089/7147648

Verk. VC 20 + 16 K + Bücher + Software (Softw. auch einzeln) z.B. Amok etc. Preis: VB DM 420, M. O. Falteisek, Am Längenbühl 3, 7253 Renningen 1, Tel. 07159/6381 ab 17 Uhr

Verkaufe: VC 20 + 3 K Supererweiterung, Literatur, Programme. Noch keine 6 Monate alt für VB 320 DM bei: Jens Helmig, Im Riephagen 26, 5000 Köln 80, Tel. 0221/681229

★★★ VC 20, 4 Wochen alt ★★★
+ Joystick + Basic-Kurs + bücher + Software (100 Programme)
ZUSAMMEN VB 220 DM, Jens Kampmann, Tel. 02853/3377

Verk. VC 20 + Datasette + Modulbox + 16 K-Karte + Grafikkarte + 3 Spielmodule + 5 Bücher
Preis 630,— DM, Interessenten ab 16 Uhr Telefon: 0202/83280

VC-20 + Modulbox + Datasette + Supererweiterung + 16 K-RAM + Maschinenmonitor + Sargon II + Literatur + Prog. (NP 1500,— DM) für 900,— DM, Tel. 0231/599999

★★★★★ Achtung ★★★★★
Suche alte oder defekte VC-1541/VC-1540. Preis VS. Angeb. an Udo Lassahn, 2725 Westerwalsede 39
★★★★★ Achtung ★★★★★

VC 20; Suche dringend Maschinensprache-Modul für sagen wir 60,—. Leider kein Telefon; Angebote an: Hubertus Sandmann, Studiostraße 8, 4800 Bielefeld 16; Rufe zurück.

VC 20 Erfahrungsaustausch VC 20 Hard-Software-Tips-und-Tricks, Utilities und vieles mehr. Schreibt an: A. Soballa, Pf.-Sansen-Str. 31, 4418 Nordwalde

Verk. komplett: VC 20!!! + 16 K + 8 K + 3 K u. Grafik + Prog. Hi. + 3 Steckpl. + 4 Bücher + Softw. viele Spiele + Schach!!!
Ab 18.00 Uhr: Köln 0221/217479

Verk. VC 20 u. 5 Steckm. u.a. Pole Position, Donkey Kong, Submarine Comander u. 2 Handbücher, NP 630 DM bei mir ca. 450 DM, Tel. 0209/28052

Defekte VC 20 + C 64
sowie Floppies gesucht. Suche Software. Schriftliche Angebote an H. Schneider, Schroverbecksweg 11, 4236 Hamminkeln 4

Verkaufe VC 20 (1 Jahr alt), mit einer Programmkassette für 140 DM, Speichererw.: 3 K, 8 K, 16 K und A000 für 70 DM. Beides zusammen: 200 DM. Tel. 0201/693003 (ab 15 Uhr).

Biete: 16 K-Erweiterung mit Umschalter auf Modulbereich ★ 100 DM. Suche: Software (GV-27 K). Nils Erben, Riekebostel 2 B, Tel. 04269/1230 ★ 2725 Kirchwaldsede

Verkaufe 32/27 KB-Modul für VC 20 für 120,— DM.
H.-Joachim Grieser
Entenpfühl 12, 5591 Faid
Tel. 02671/8905

!!! Wanted !!! Wanted !!!
Wer tauscht mir sein 27/32 K-Modul gegen meine 16 K + Aufpreis (nach Vereinbarung)??? Eilt: Meldet euch bei Udo Lehmann unter 0231/370190

Verk.: VC 20 o. HF-Mod. 100 DM, 64 K-RAM 50 DM, 40/80 Z. 150 DM, VC 1515 50 DM, Mod. Box 200 DM, Datasette 50 DM, 16 K 80 DM, 3 K 20 DM bei J. Hutflies, Erlenstegenstr. 124, 85 Nürnberg

Ich baue Ihre Original Commodore 16 K-Erweiterung in eine schaltbare 16 K-Erweiterung um. 20 DM und Erweiterung an E. Daamen, Norstr. 14, 4155 Grefrath 1. Umgehend zurück.

Verkaufe VC 20 + Modulerw. 25 Lots + 3 KB + Grafikerw. + Basickurs + 3 Bücher + Datasette + Joystick für 400 DM. Thomas Döpke, Am Sandbrink 9, 3220 Alfeld, (05181) 6892

Achtung!
Verkaufe VC 20 + Datasette + Handbücher + Software auf Kass.
Tel. 08041/6644 (ab 18 Uhr)

★★★ Geschenkt ★★★
bekommen Sie neuw. VC 20 + Datasette + Joyst. + Literatur + Softw. + Tasche nicht! Aber fast! 350 DM. F. Pietsch, Grundbirmgarten 5, 6780 Pirmasens

Verkaufe: VC 20 + 200 Programme + 2 Computerbücher ohne Datasette für: 200 DM, noch kein Jahr alt, schreibt an: Klaus-Dieter Behnke, Krögerstr. 73, 3300 Braunschweig

Verkaufe VC 20 + Datasette + 16 K Erweiterung + 21 Kass. voller Superspiele + viel Literatur für nur 398,— DM. U. Fischbach, Tel. 0228/654134

Suche Spiele und andere Programme für VC 20 — 16 K Erweiterung. Guido Wippler, 4130 Moers 2, Ringstr. 48, Tel. 02841/63612

● ● ● Verkaufe C-64 und VC 20 ● ● ●
Mal was anderes, verkaufe HSV-Spiele auf normaler Kassette!!! Angebote an Frank Zingelmann, Habichthorst 42, 2000 HH 61

★ Verkaufe ★: Orig. Commodore 16 K Speichererw. (VIC 1111) — fast neu — wegen Systemwechsel. Preis: VB 125 DM. ★ Stefan Munsch, Unterbüngstr. 53, 5450 Neuwied 22 ★

★ V-E-R-K-A-U-F-E ★★ V-C-2-0 ★
VC 20 + 16 K + User-Port-Experimentierplatine + div. Software, meistbietend zu verkaufen
06130/5427 (17-19 Uhr)

Suche günstige 80-Z-Karte f. VC 20 und Monochr. Monitor sowie (auch defekt) C64, 1541 od. 1540 Postkarte bitte an H. D. Schauer, Frankenring 74 A, 8555 Adelsdorf

COMPUTER-MARKT

Private Kleinanzeigen

Verkaufe: VC 20 + 16 K-Speicher + Datasette + Prgs. (Supertape) für nur 300,— DM!!! Zu melden bei Michael Jaumann, Augustenstr. 17, 8958 Füssen.

Suche Commodore VC 20 mit Zubehör. Telefon 06 51/1 08 09

Verkaufe VC 20-Computer mit 2 Modulen und 3 Kassetten und Handbuch für 300 DM. Mark Lohmann, Dr. Deistingstr. 10, Kierspe, Tel. 0 23 59/33 04

Verkaufe VC 20 + 3 K Erw. + 32 K Erw. voll schaltb. + Schachmod. + Bücher + ca. über 100 Spiele + defekte Dataset. VB 420,— DM. Joachim Vogt, Mainzer-Allee 53, 7140 Ludwigsburg

Verkaufe VC 20 + 3 K + 8 K + 16 K + Software + Literatur VB: 350 DM. Frank Müller, Frankfurterstr. 44, 6057 Dietzenbach, Tel. 0 60 74/3 19 72

Suche für VC 20 (Tel. 0 81 57/81 26) Modulbox, 3-5 Steckplätze, Module: 3 K-Erweiterung, Masch.-Monitor. Angebote an A. Sperber, Possenhofener Str. 6, 8133 Feldafing

Suche Spiele (Kassette), Bücher, Speichererweiterung, Akustikkoppler, Programme aller Art, Listen, Angebote an M. Kober, Dorfstr. 3, 6483 Bad Soden-Salmünster

Suche für VC 20 (GV) Software-Programme, vor allem Adventures, m. u. o. Grafik (auf Kass.). Angebote m. Preisvorstellung: J. Gerschütz, Kirchtorstr., 8721 Stadtlauringen

VC 20 + schaltb. Mod. Box 1020 mit 6 Slots + 16 K-RAM + Datasette + Resettaste + Joystick + Lit. u.v. Software (u.a. Basic-Kurs) für 680,— DM zu verk. (NP 1 150,—), Chr. Emrich, Telefon 0 76 22/77 80

■■■■■■■■■■■■■■■■■■■■
!Verkaufe!
VC 20 + 16 K + Datasette + Literatur + ca. 100 Progr., VB 400 DM; Melden bei: Michael Schäffer; 0 83 62/8 13 07

Verkaufe VC 20 und Software sowie 3 K-ROM Super Erweiterung, Telefon 0 22 63/60 54

Verkaufe VC 20 + 32 K-Erweiterung (schaltb.) für nur 320 DM. Suche C64 bis 500 DM. Angebote an Alexander Knöller, Bahnhofstr. 14, 7545 Höfen/Enz

Verkaufe: — VC20 — (leicht beschädigt) + 1 Buch + 20 Spiele VB 150 DM!!! — Philips Masterlab + Zubehör Top-Zustand, NP 450, VB 200 DM! Marco Zens, Steinweg 2 a, 5330 Königswinter 41

★ VC 20 ★
Verkaufe VC 20 (4 Monate alt) mit Datasette + Software + Handbüchern für 260,— DM, Tel. 0 42 02/7 10 47

Hallo VC 20-User! Suche billige 32 K Erw. (ca. 50 DM). Tausche auch Prg. (GV-16 K u. A$000). Liste an Weber Alexander, v. Parsevalstr. 44, 8900 Augsburg 1, Tel. 08 21/57 24 39

VC 20 + 40/80 Zeichenk. + Superexp. + 16 K (schaltb.) + Literat. + Prgme. komplett 500,—. Suche Floppy und Tape, a. Tausch. Cramer, Lahnau 1, Lauterstr. 43, 0 64 41/6 28 73

Verk. VC 20 + 3 K + 16 K + Tool u. Abs. in Floppy Geschw. + 3 Pl. Box + 40 Zeichen pro Zeile u. 25 Zeilen u. 20 BSP. Prg. + 40 Prg. (Spiele, Scrool) = 350 DM • Auch einzeln • Anfragen: 0 27 71/2 25 95

★★★ Suche für VC 20 ★★★
Supererweiterungsmodul, Schach- und ähnliche Spielprogramme
Harald, Sünderwald
Kärntner Str. 10, 7250 Leonberg

VC 20: Suche dringend Forth-Modul für sagen wir 60,— DM. Angebote an: H. Sandmann, Studiostr. 8, 4800 Bielefeld 16. Ich rufe zurück.

★★★ Suche für VC 20 ★★★
Erweiterungen 16 K und größer Modulbox, Joysticks, Lichtgriffel
Harald Sünderwald
Kärntner Str. 10, 7250 Leonberg

Drucker für VC 20/C64 Seikosha GP 100 VC wenig gebraucht gegen Höchstgebot abzugeben. Dieter Krause, Seestr. 47, 6050 Offenbach, Tel. 0 69/86 34 73

Verkaufe für Commodore VC 20:
Dateiverw.: VIC-File nur DM 70
Program.-Hilfe: VC 1212 nur DM 50
Versand geg. Verrechnungsscheck, J. Jahn, Rotkreuzstr. 33, 858 Bayreuth

Orig. Modulbox + VC 20 (nur zus.) 16 K, Supererw., Progr.hilf, 2 Spielmod. + div. Prg. Preis: n.V.
H. Hötger, Mayrhofener Weg 27, 6380 Bad Homburg, Tel. 0 61 72/4 58 1 45

VC 20 + 32 K + 40/80 ZK + Modulbox + 3 K + Datasette + Drucker VC 1515 + Software wegen Systemwechsel zu verkaufen DM 700,—.
0 93 43/51 54 ab 17.00 Uhr

■ **VC 20** ■ **Achtung!** ■ **VC 20** ■
100 Pgm.-Kassetten (alle in C-Box!) — Superspiele + Utilities! — wegen Syst.-Wechsel für DM 200,— z. verk. T. Nitz, Krablerstr. 24, 43 Essen 12

Preiswert zu verkaufen:
VC 20 = 224 DM/Datasette = 89 DM/3 K = 36 DM/16 K = 85 DM/RTTY f. VC20 = 339 DM.
Alles Neuwert/H. Schneider
Tel. ab 17 Uhr 0 28 57/23 36

Drucker für VC 20/C-64 Seikosha-GP 100 VC sehr wenig benutzt, neuwertig gegen Höchstgebot zu verkaufen. D. Krause, Seestraße 47, 6050 Offenbach, Tel. 0 69/86 34 73

VC 20 (1a Zustand) + Supererweiterung + ca. 100 Spiele (Centipede, Scramble, ...) + Bücher für 190 DM. Stefan V. Campe, Tel. 07 11/76 48 42

VC 20 Infoblätter mit vielen Tips, Probe: DM 5 (Schein) — Alle: DM 20 (Schein)
Erlös für Äthiopienhilfe
(Menschen für Menschen)
Engbrinkkamp 24, 4432 Gronau

VC 20 + Datasette + 16 K-Speichererw. + ca. 40 Spiele + Spielmodul + Joystick für DM 420,— zu verkaufen. Tel. (0 69) 78 27 96

Verkaufe VC 20 + VC 1020 + 27/32 KB + Datasette + Joy. + Liter. + Software + Reset. VB. Telefon 0 23 07/6 16 07 14 h bis 20 h, Marc Engelbert, Pfalzstr. 79, 4709 Bergkamen

VERSCHIEDENES

Suche für C64 auf Disk Compiler/Interpreter von: Pascal, Forth, Fortran, Cobol, Basic, Simons Basic. Liste an: H. Fuhrmeister, Postfach 1350, 6090 Rüsselsheim

SUCHE
Heft Nr. 4 + 5 von »64'er« zu kaufen oder leihweise zum Kopieren. Angebot an: H. Fuhrmeister, Postfach 1350, 6090 Rüsselsheim

■■■ LINK 64 ■■■
Modem Akustik Koppler Modem Komplt., Anltng., Muscheln, Betriebsbereit, Fr. 220,—, PF 315, CH-8042 Zürich
■■■■■■■■■■■■■■■■■■■■

Schüler sucht DEFEKTE HARDWARE jeder Art gegen angemessene Bezahlung (Computer, Floppy, Module, Drucker ...) Tel.: 07 11/26 11 77 T. Ostmann, Raitelsbergstr. 45, 7000 Stuttgart 1

★★★★ **Original-Disketten** ★★★★
Simon's Basic 95,— DM, Quickcopy V 2.0 30,— DM, Epson Interface Disk 25,— DM, jew. Orig. Beschreibg., J. Knöpke, Gartenstr. 10, 3339 Söllingen

C-64, VC 1541, 10 Disketten wenig gebraucht, 3 Monate alt, wegen Zeitmangel zu verkaufen, 1 200 DM
Tel. 0 76 27/73 71

★★★ Verkaufe für VC-20 ★★★
Modulbox VC-1020 DM 100
16 K DM 100 Prg. Hilfe DM 25
8 K DM 50 Masch. Mon. DM 25
div. Bücher Tel.: 06 21/85 27 23

Suche Floppy VC 1541 bitte keine Apothekerpreise!
Schallplattenarchiv auf Kassette, Angebote an Gerd Brechtel, Plaumfeld 47, 8820 Gunzenhausen

Neu! Nutzung der Diskettenrückseite ohne Probleme mit der **Disk-Doubler** Kunststoffschablone! Funktioniert schnell, sicher und einfach, einlegen, lochen, fertig! **Nur DM 10,—** (Schein) oder NN. User-Disk-Service, Postfach 34, 3014 Laatzen 4, Tel. 0 51 02/41 36

CBM-Floppy 2031, schreib- u. lesekompatibel zur Floppy 1541, 6 Monate alt, für 900 DM zu verkaufen.
Tel. 0 23 33/8 02 02

Generalüberholter Fernschreiber mit Lochstreifensender u. -empfänger für 85 DM zu verkaufen. Tel. 0 23 33/8 02 02

Texas TI 99 Suche günst. Computer + Zubehör (Hardware)/Angeb. bitte schriftl. an L. Bialasinski, Leipziger Str. 6, 4030 Ratingen; Melde mich sof. zur.

Christiani Basic Kompaktkurs NP 200, Lösungsaufgaben wurden nicht beschriftet! Wie neu! Mit Original Tonkassette nur DM 150/inkl. NN. G. Krause, Schlesierstr. 3, 877 Lohr

Hallo Nena, --------
ich habe von einem Magazin gehört, das SOURCE heißt und nur über COMMODORE schreibt. weißt Du wo es SOURCE gibt?
viele Grüße von Desiree -------

ZUBEHÖR

Verkaufe: Görlitz-Interf. (neueste Version) DM 280 — 5* Centronicstecker + Kupplung (NP DM 40) 2* Quickshot I je DM 19 — **Suche:** gebrauchte Eproms, T. 0 72 46/13 40

Verkaufe: Tausche 64er Prg....
Disketten 10 Stück 60,— DM
Nagelneu Formatier und Laufgepr. Rücksend-garantie/SS/DD/ T.H. Soft, Jakobstr. 4, 7317 Wendlingen

Suche Turbo Access oder Speeddos, das Buch Programmierung des 6502 von R.Z. und das PRG Grafik-Basic von Hesware (auch Raubkopie + Anleitung) ab 13.30, 02 01/51 33 39

■■■ Endlich! ■■■
Modem Gehäuse für Bausätze, K. Schubert, Rottmannstr. 10, 6900 Heidelberg, 48,50 per NN
■■■ inkl. Porto u. Verp. ■■■

Tausche Printer/Plotter VC 1520 gegen Grün-Monitor. Suche defekte Floppy 1541 und suche deutsche Anleitung zu Vizawrite und Vizastar ab 18 Uhr, 0 23 77/73 47

■■■■■■■■■■■■■■■■■■■■
Suche Centronics und RS232 (Schnittstelle)
Tel. 0 69/41 89 66
■■■■■■■■■■■■■■■■■■■■

★★★★★★★★★★★★★★★★★
Drucker VC-1515 mit Papier gegen Höchstgebot, mind. 230,—
F. Schnell, 4c-2-216, 8014 Neubiberg
★★★★★★★★★★★★★★★★★

VC 1530 Datasette zu verkaufen; fast wie neu!!! VB 60,—, Marc Kurzmann, Jahnstr. 21, 8210 Prien/Ch. Tel. 0 80 51/33 87

■■■ Floppy gesucht ■■■
Suche defekte VC 1541. Angebote mit Preis und Schadenbeschreibung bitte schriftlich an Ch. Wittmer, Adlismattstr. 6, CH-8905 Arni

★★★ **HILFE!!!** ★★★
Suche dringend gebrauchten **C16!!** Möglichst mit Software. Melden bei: Bernd Klimkeit, Nelkenweg 23, 5308 Rheinbach, ☎ 0 22 26/1 28 91

Verk. Atari VCS + 18 Kassetten für 900 DM sowie 16 G7000 Kassetten für 700 DM / Suche auch C64 Club im Raum Ffm sowie Software (Kassette) Gerd Viol, Tel. 0 61 71/7 90 90

Verkaufe elek. Schreibmaschine (Brother EP 22) mit Netzteil und Interface für C64. Preis: 500,— DM, 3 Mon. alt, Neupreis: 700,— DM.
K. Vogel, Jägerstr. 5, 3062 Bückeburg

Suche Controller-Platine für Floppy 1540/1541 oder def. Floppy, Dieter Glenk, Tel. 0 96 45/16 42

Fernschreiber LO 133 inkl. Interface und Treibersoftware für VC20 und C-64 für 250 DM. Topzustand weniger 1 000 Betriebsst.
Tel. 0 91 01/80 01 nach 17.00

Printer EP 22 + Anschlußkabel für C64 — originalverp. — H. Weikert, 0 61 57/8 18 07 abzug. f. DM 240,—

Verkaufe Printer/Plotter 1520!! Nur 3 Monate alt, 1a Zustand mit Zubehör, für nur 350,—. Bitte melden ab 17 Uhr bei Tel. (0 54 66/4 19)

Verkaufe Drucker HR-5C für C-64 von Brother originalverpackt mit Garantie für 439 DM. H. Masuch, Bahnhofstr. 24, 6293 Löhnberg 1, Tel. 0 64 71/6 11 19 ab 20 Uhr

BTX-Modul für C-64
Suche für C-64 BTX Modul neu o. gebraucht.
Roland Rosner, Weißenburger Str. 16, 8803 Rothenburg

Verk. Resettaster VC20/C64 Stück 5 DM inkl. Porto! 5 Stück 20 DM + 1,10 Porto! Kein Löten! Nur einstecken! Geld in Vorkasse an A. Jacobsen, Norderf. 1, 2267 Weesby, 0 46 05/8 61

Verkaufe VC 1530 Datasette 1a Zustand m. Bed. Anleitung
Preis VB
Tel. 0 81 42/1 40 96

★★★★★★★★★★★★★★★★★
Verkaufe Matrixdrucker VC
ginalverp. + Handbuch, 1
DM, Tel. 0 93 31/608 (N. 1
★★★★★★★★★★★

Verkaufe Atari-Telespiel 1
Drehregler + Tastatur, 1
750 DM. Comos CP 1 C
VB 200 DM. Tel. 05
18.00 Uhr

Neu — und einmalig von
unentbehrlich für Prog
Dezim-Binär-Rechensc
DM im Brief. Tel. 02 34
Bochum 7, Hauptstraß

Private Kleinanzeigen

C-64 Frequenzzähl-Steckmodul bis 30 MHz, Eingangsempfl. 50 mV + Software auf Kass. o. Disk: 50 DM, R. Kvapil, Heimgarten 6 a, 3300 Braunschweig, Tel. 0531/322362

Verkaufe original Multiplan mit Handbuch für 180,— DM.
Suche Musi-Calc, suche Musi-Calc, suche Musi-Calc, 0681/512 94
★★★★★★★★★★★★★★★★★★

Suche
Bücher zum 64, (auch Kopien, Anleitungen) und Hardware. Angebote an: M. EGGER (a. Tausch!), Schalksberg 7, 6694 Urexweiler

3,— DM ★ 3,— DM ★ 3,— DM ★ 3,— DM für 3,— DM Vorkasse gibt es bei mir einen Resetstecker für den VC 20/64: Jürgen Heinert, Hans-Seibold-Str. 7, 8950 Kaufbeuren

★★★ Achtung C 64/VC 20! ★★★
Resettaster inkl. Renew je 7 DM. Ab 3 St. je 5 DM. Joystickdauerfeuer je 20 DM. (Vorkasse — Bfm/Schein) Roland Evers, Stifterstr. 15, 4020 Mettmann

Suche Interface — Biete Epson RX80-FT & VC 64, Bargeld
P. Möller, Dorotheenstr. 5 A, 3 19/3 H 21

Userport-Kabel 1m-Rundkabel mit Stecker im Gehäuse 15,— Scheck an Hass, Hochstr. 38, 4000 Dssdf. 13

★★★★★★★★★★★★★★★★★★
Je 1 **CBM-Printer 4022P und 4023** zu verkaufen. Zustand wie neu, Preis Verhandlungssache
★★★ 0931/49200 ★★★

Reset-Taster für C-64, VC-20, kein Löten! Bei Nichtfunktionieren Rückgaberecht, 10 DM an: Volker Debus, Hartenroder Str. 5, 3551 Bad Endbach 1

C-64 Reset-Taster-Stecker!!!
Kein Datenverlust bei Systemabsturz, inkl. Re-New-Programm, kein Löten, kein Garantieverlust, 10,— DM, Tel. 02331/51807 ab 19 Uhr

■■■ !!Dringend!! ■■■
Suche gebrauchtes Floppy 1541
■■■ **Zahle bis zu 450 DM!!** ■■■
■■■■■■■■■■■■■■■■■■■■■
Interessenten an: Tel. 06196/1242

Hardwareerweiterungen, Schaltinterface 220 V, Lauflicht, Autostart Eproms, laden nach dem Einschalten direkt PRG von Disk Info 1 DM, Lutz, Mannheim 31, Lemaitrestr. 10 F

★ Neu ★ Neu ★ Neu ★ Neu ★
60 Min. Musik, gespielt vom C64 & Demo Sprachsynthese. Stereo + Dolby Kassette für DM 20 (Schein) bei J. Renfer, Ebertstr. 43, 67 Lu-Hafen

1541 Floppy 1541
Suche Floppy 1541. Zahle je nach Zustand bis 500 DM. Floppy muß technisch o.k. sein. Angebote an M. Neubauer, n. 17 U., T. 02605/3537

★★★ **Achtung C64 User** ★★★
Wir brennen Eure Software auf Eprom. Leere Eproms + Epromsteckkarten erhältlich. [Typ 2716-27128], günstige Preise, Tel. 0209/788543

Suche für C64 Korresp. Drucker (Matrix/Typenrad), 80 Zchk. Monitor 1541; Modem. Biete VC1515 250,— DM. St. Birner, Rhönring 15, 6100 Darmstadt, 06151/784635

★★★ Eprom-Brenner ★★★
Verkaufe Zero-Eprombrenner kompl. mit Stecksockel u. Treibersoftw. für 170 DM.
Tel. 07144/4851 (19-21 h)

Farb-Monitor gesucht. Um 300 DM. Angebote bei Andreas Mattern, 6700 Ludwigshafen, Tel. 0621/689920

Profi Disc Box, verschließbar 30 DM, Reset-T., ReNew gratis, k. Löt. 8 DM, Lightpen 50 DM — Schutzhaube für VC 64, Floppy, Tape je 12 DM. Scheck an: Awad, Lützowst. 80, 46 Dortmund 1

Kabelsaat auf dem Schreibtisch?
Lösung: Tastatur und Computer trennen. Ausf. Bauanleitung ink. Reset nur 20 DM. VR-Scheck an Trümper, Sendst. 133, 46 Dortmund 1

! Umsonst !! will ich nichts. Aber eine gebr. eventuell defekte !! Floppy !! Die billigste wird gekauft von Oliver Schindelbeck, 7297 Alpirsbach, Dieboldsberg 26

Biete an: 40/80-Zeichenkarte für 130 DM; Supererweiterung-VC1211 A für 80 DM sowie Steckplatzerweiterung auf 5 St. Plätze 30 DM, alles zusammen 210 DM. Tel. 02233/46861

Verkaufe Simon's Basic Modulversion + Data Becker Handbuch, beides neuwertig und originalverpackt. 130,— DM, VB. Jens Eschrich 05161/2377

Basiclehrgang »Gortek und die Mikrochips« + 2 Kassetten und Handbuch. 30,— DM. SYBEX-Programmsammlung, Buch, neuwertig. Alles originalverpackt 20,— DM. Jens Eschrich 05161/2377

Light-Pen für C64 mit Software-Programm auf Disk (Mal/Zeichen u. Spielpr.) für DM 50,— zu verkaufen. Tel. 02161/641224 nach 18.00 Uhr.

Suche VC 1520
Guth, Böhmerwaldstr. 25, 8300 Altdorf

Seikosha GP 100 VC, anschlußfertig für VC 20/C64, wenig gebraucht DM 400,—, Tel. 02362/75839

Eprom Platine für zweimal 2764 oder 2732 als 2x 8 KB oder 1x 16 KB-Modul einzeln schaltbar mit Umschalter, Resetschalter fertig bestückt 29 DM, 02381/71008

! Modemplatine (fertig) nur DM 35 ! entspricht CCITT Norm V.21. (Bauanleitung aus: Modem-Sonderheft) mit Schaltplan + Anleitung..
Call: A. Steppat, Tel. 07123/71815

Diskettenlocher 15 DM — Resettaster 8 DM — Disketten ab 490 DM — Gratisinfo — C — anfordern von H. Masuch, Bahnhofstr. 24, 6293 Löhnberg 1, Tel. 06471/61119 ab 20 Uhr

Hilfe!! Suche dringend Beschreibungen für Proplan, Magic-Desk I und Data-Manager 64. Suche Tauschpartner im Saarland. M. Weber, Meerwiesertalweg 10, 6600 Saarbrücken

★★ Commodore-User aufgepaßt!! ★★
Resettaster, Floppykabel, Monitor- und Fernsehkabel, Reseteinbau in C 64 und VC 20: Info bei Teddy-Soft, R. Rahn, Hauptstr. 180, 5102 Würselen

Wer verkauft (verschenkt) zu Niedrigstpreis: eine Floppy 1541; einen Drucker (z.B. VC 1520); und einen Grün/Farbmonitor für C64 an Schüler? J. Seidl, Im Gillesbachtal 31, 5100 Aachen

■■■■■■■■■■■■■■■■■■■■■
VC-1520 VC-1520
6 Monate alt, Gerätenummer = 4
Preis VS, Tel. 04161/2324

Brother-EP 22, Schreib-Printer — RS232-Schnittstelle — 17 Zeichen/Sek.-Netzteil, 1 Jahr alt.
VB 400,—
Tel. 05971/15244

Floppy 1541/40 Schreib-Lesekopf-Justierung in etwa 10 Min. ausführl.! Anleitung gegen 10,— DM an: H. J. Kuckertz, Im Hag 34, Tel. 02403/28151 ab 19 Uhr!

Kabel Userport nach Centronics, Länge 1,5 m DM 55 + NN DM 5, Treiberprogramm auf Anfrage
VC-User
Postfach 1433, D-6908 Wiesloch 1

Verkaufe Floppy 1541, 4 Mon. alt und Akustikkoppler Epson CX-21 + Teleterm + Schnittstelle für C-64, 4 Woch. alt gegen Höchstgebot. Verkauf der Teile auch getrennt. 02369/8862 ab 19 Uhr

Suche Interface, Kabel + Software zum Anschluß einer Brother/CE 60 Typenradschreibmaschine an C64. Angebote an Peter Jarzabek; Hölderlinstr. 32, 4250 Bottrop

Verlängerungskabel
für den seriellen E/A-Port
2 m Stecker-Stecker 10 DM
5 m Stecker-Stecker 20 DM
+ 5 DM Vork. Ingenhorst, Jahnstr. 19, 33 Braunschweig

Reset-Taster
f. den seriellen A/E C-64/VC-20 5,— DM Vorkasse///Harald Krist, Postfach 3152, 5205 St. Augustin 3,
Suche für C-64 Schaltpläne

Suche Schaltpläne für C 64 und Peripherie aller Art (Modem usw.). Hallo C64-Freaks aus Landkreis Miesbach und Umg. Plane Interessengemeinsch. Ab 19 Uhr 08022/81732

Verkaufe VC1520 Printer-Plotter, 5 Monate alt, VHB 350,— DM
Tel. 040/7135267 M. Lilienborn, Wiesenweg 28, 2000 Oststeinbek

Wer kann mir für ca. 50 DM eine Sprachausgabe (f. C64 m. Demo) bauen o. den Chip **SP 0256-AL2** besorgen? Angebote an: Bernhard Rabiser, Rauschbergstr. 19, 8221 Inzell

Farb-Foto-Labor m. Chemie-Papier, Coloranalyser + Literatur, kompl. mit VC 20 + C64 steuer- + regulierbar. Kompl.-Preis 2500 DM: Mertes W. 5532 Jünkerath. Pf. 72, 06597/2991

★★★★ **Achtung!** ★★★★
MPS-801, 4 Mon. kompl. m. Farbband-Renewset: Mertes W., Bahnhofstr. 46, 5532 Jünkerath. Tel. 06597/2991
★★★★ 550,— DM ★★★★

★★★★★★★★★★★★★★★★★★
Student sucht Z80-Karte mit Anleitung. Kaufe gebrauchte Datasette und defekte Floppies.
Tel. 09132/5409
★★★★★★★★★★★★★★★★★★

★★★★★★★★★★★★★★★★★★
Tandy-Koppler AC-3 + Schnittstelle (FTZ-Nr.) wenig gebraucht, 450 DM — Lauer → → → 0228/213785
★★★★★★★★★★★★★★★★★★

★★ Achtung Koppler-Freaks ★★
Robustes, zweckmäßiges Holzgehäuse (Bausatz) mit Tel.hörer, Abdichtungen für Selbstbaukoppler. 50 DM inkl. Porto. Tel. 07457/1245

Farbmonitor für C64 preisgünstig zu kaufen gesucht. Tel. 07151/54748

■ **64'er** ■
■ Alle Ausgaben. ■
■ Auch Einzeln. ■
■ Geg. Gebot ■
■ Tel. 089/160698 ■

Suche für Seikosha GP 100 Mark II Interface an C 64 (Bauanleitung, Bausatz oder günstiges Fertigteil). Dringend!!! Helmut Strahl, 4223 Voerde, Teichacker 12, Tel. 02855/7629

Zusatztastatur C64/VC 20 keine Software nötig, 16 Tasten, Anschluß ohne löten!! Einzelteile zum Selbstbau. Telefon 07625/7147 abends

Original Görlitz VC-Epson-Interface für DM 280,— zu verkaufen. Telefon 0281/43568 nach 18 Uhr

Verkaufe Prommer von Roßmöller, aufgebaut, getestet, im Gehäuse (alte Version) 20er-Softw. 200,— DM VB. B. Leikauf, Wuhrmatten 17, 7836 Bahlingen

C-64. Verkaufe Centronics-Interface, 200 DM ■ 80-Zeichen-Grafik-Karte (Org. Roos 2 Mon.) 288 DM ■ R.B. Gansel, Kammerstr. 100, 4100 Duisburg, Tel. (0203) 357051

★★★★★★★★★★★★★★★★★★
Data-Becker-Interface für FX 80 oder RX 80, VB 200 DM. Tel. 02325/73540 ab 19 Uhr.
★★★★★★★★★★★★★★★★★★

Suche folg. gut erhalt. Data Becker-Bücher: Adventures, Lexikon z. C64, Floppy-Buch, Grafikbuch z. C64 ★★★ Angebote an: U. Schwenke de Wall jun., Harzweg 8, 338 Goslar 1

Görlitz Druckerinterface Testsieger in 64'er 7/84 250 DM. Suche gute Anwendersoftware. M. Brandenburg, Heitmannsweg 3, 2803 Weyhe, Tel. 04203/826

★ **Flugsimulator** ★ Ausf. deutsche Anltg. in 1a-Qualität inkl. 4 Flugkarten mit 79 Flugpl. für 30,— (Schein/Scheck) J. Sonneborn/Postf. 130736, 56 Wuppertal 1
★★ Tel. 0202/732439

■■■■■■■■■■■■■■■■■■■■■
Diskmonitor, Disk-Tape-Backup, Directory-Sortierprogramm, Programmdatei, Info gratis bei: B. Natter, Wallbergstr. 14, 8011 Höhenkirchen
■■■■■■■■■■■■■■■■■■■■■

Verk. Sprachmodul für C 64, DM 148, Userport-Anschluß, beliebiger Wortschatz, mit ausführl. Manual, H.-J. Schiller, Tulpenstr. 17, 7556 Ötigheim

■ **12 Bit A/D-Wandler** ■
mit Software und Beschreibung für C 64 als anschlußfertiges Modul nur 80 DM, Dipl.-Ing. Merkl, Innsbruckerstr. 47, 7100 Heilbronn

Userport-Tastatur, 20 frei programmierbare Tasten, anschlußfertig im Gehäuse, für nur 120,— DM, Info gegen 80 Pf. Briefm. bei Detlef Link, Piwipperstr. 17, 4047 Dormagen

★ Drucker ★ Microline 80, 7 x 9 Matrix, Einzelbl. + Traktor möglich. Interface für VC 20 + C64 vorh. R. Jakubeit, Lenbachstr. 26, 5650 Solingen, Tel. 02122/311862

Gehäuse für Akustik-Koppler inkl. Gummikappen 69 DM kompl. mit Elektronik 300 BD/1200 BD Orig./Ans. 298 DM, Edgar Bockhacker, Kastanienallee 15, 3000 Hannover, Tel. 0511/835315

★★★★★★★★★★★★★★★★★★
Fernschreiber als Drucker
Am C 64. Info gegen Freiumschlag. H. Quante, Postfach 11, 4714 Selm 2
★★★★★★★★★★★★★★★★★★

MPS-802: Hardcopy-Programm für Disk, 100% MCode, 1,2,4facher Hires-Ausdruck aller Bilder, Super-Editiermodus, 20 DM an E. Stolle, W. B. Feldstr. 51, 7082 Oberkochen

Probleme mit dem 1526? Kein Zeilenvorschub bei undefinierten Zeichen? Zeichensätze: dt. oder Schreibschrift. Info bei M. Böhmel, Am Töbele 2, 7923 Königsbronn

OLYMPIA Typenradschreibm. report electronic anschlußfertig an C-64/VC 20 inklusiv gute Softw., 1a Maschinen, das Textsystem!! DM 1095,—, Schmidt, Tankredstr. 5, 1 Berlin 42

SANYO-Monitor (grün) neuwertig für 200,— DM zu verkaufen. 2 Monate Garantie noch vorhanden. Tel.: Krefeld (02151) 795340

Verkaufe Spreizkartei für 20 (!) Disketten DM 25,—, Reset-Taster DM 10,—! **Suche** Joysticks! Oliver Eichhorn, Rübekamp 10-12, 28 Bremen 1, Telefon 0421/6166486

★★★ Suche ★★★
Commodore SX 64 im Tausch gegen CP/M 64 + VC 1541 + VC 1530 + 30 Disketten, Software + Bargeldausgleich, Tel. 09262/222 ab 18 U.

Suche dringend Akk. u. Modemschaltpläne + Software (C64 + Rec.), Kopien bitte an Christian Michlmayr, Landhauspl. 2, 6020 Innsbruck, Österreich

Huckepackplatine: 1526 mit zweitem Betriebssys. Modifiziertes Betriebssys. für C64. Preis 1526 = 60 DM, C64 Betr. = Infos gegen Porto, M. Schildmacher, Bahnhofstr. 12, 4840 Rheda-WB, Tel. (05242/48898)

Der Drucker ohne Probleme: MPX 802 Commodore. Endlospapier und Einzelblatt mit Zubehör. 5 Commodore-Drucker zum Spottpreis von 750,— inkl. MwSt. zu verkaufen! Greifen Sie zu! Tel. 04533/2431

★★★ C 64 ★★★ C 64 ★★★
Wir brennen Ihre Prg. in EPROMs! Steuerkonsolen, Disketten, Res.-Taster, etc. Gratis-Info auf Anfrage. Tel. 0234/581875

VC-1540/41: Lesekopf in ca. 10 Min. selber eingestellt! Einstellanl. gegen Einsendung von 10 DM (Bar, Scheck o.ä.): Volker Mücke, Im Hag 32, 5180 Eschweiler//**Sofortversand**

STAR Gemini 10 X Drucker mit VCCI intelligentem Interface (Umlaute, Hires-, Comm.-Blockgrafik) neu, mit Garantie + 1 Kiste Papier, VB 1099 DM, Tel. 06106/71689

Wertvolle VC-20/64 Tips & Tricks
Sofortige Lieferung gegen je 25 DM — Vorauskasse — Info gegen —,80: Volker Mücke, Im Hag 32, 5180 Eschweiler // Es lohnt sich 100%ig!

Turbo-Tape Steck- ★★★★★
★★★★★ module **Turbo-Disk**
Bis zu 10x schnelleres Laden von Kassette und Diskette!!!
SUPER PREISWERT!!! 02151/27267

Am 20.12.1984 kostet mein KFC-80-Zeichen/Grafikmodul, drei Steckpl. 2716/2532 frei, Reset-Taster, 80 DM, dann jeden Tag 1 DM weniger. Nur 1 vorhanden. Tel. 09286/1222

Wegen Systemwechsel diverse Soft- u. Hardware f. VC 20 und C-64 zu verkaufen. Liste 80 Pf. R. Bärwolf, Postfach 100321, 4650 Gelsenkirchen

VC 64/20 Akustikkoppler
Nur 98,— DM
Fertigplatine für Userport, Marcus Weber, Gülthausstr. 7, 7129 Brackenheim 2, Tel. 07135/8411

Verkaufe Seikosha GP-700 A mit Microscan-Interf. ½ Jahr alt! Preis: ca. 800 DM, oder?! Melden ab 18 Uhr bei: Christian Gohlke, Soltaus Allee 6, 2 HH 71, Tel. 6914821

Verk. Recorder-Interface, NP: DM 45, VB: DM 20, Jürgen Götz, Borsigstr. 21, 6054 Rodgau 3

★★★ Die Mailbox für den C 64 ★★★
zu verkaufen
mit Koppler Epson CX-21, mit Orig. Teleterm und RS232, mit Automodem und Wählautomat, mit Mailboxsoft sofort Betriebsbereit und auf alles noch 4 Mon. Garantie »Preis VHB«
Tel. 040/73113488 od. in einer der HH-Boxen für CATWIESEL

★★★★★★ Achtung ★★★★★★
Alle Ausgaben der 64'er!!! Heft 4/84 bis 11/84 nur komplett zu verkaufen. DM 50,—. Tel. 0541/79522

Suche Drucker Seikosha 550 A mit Schnittstelle zum Commodore 64. Bitte Angebote an: Jürgen Hutflies, Erlenstegenstr. 124, 8500 Nürnberg 20

Spricht Ihr Computer nicht mit Ihnen? — Nein? — dann wird es aber Zeit, daß Sie sich einen Sprachsynthesizer anschaffen! Noch Fragen? Rufen Sie an! 02151/27267

GEWERBLICHE KLEINANZEIGEN

++++++ OHRTRAINER +++++
Die professionelle Gehörschulung für C 64. Info gegen 2,— DM.

ohrtrainer mini	C 59,—/D	69,—
ohrtrainer 1	C 89,—/D	99,—
ohrtrainer 2	C 109,—/D	119,—

Wilhelm Lutz Verlag
Postf. 110749, D-61 Darmstadt

★★★★★★★★★★★★★★★★★
★ **Software + Hardware + Zubehör** ★
★ **für C-64 + CBM 3032 bis 8296.** ★
★ Den neuen Katalog fordern Sie am ★
★ besten gleich mit Briefmarken an ★
★ (DM 1,50). Es lohnt sich sicher! ★
★ E. Zeller, Pf 1, 8901 Batzenhofen ★
★★★★★★★★★★★★★★★★★

COMPUTER-MARKT

Private Kleinanzeigen

!! V24-INTERFACE !! Kabel und Stecker für Userport und Akustikkoppler. Preis DM 79,—
!! FÜR EUREN AKUSTIKKOPPLER !! TPMON 64-V24 — Die Komplettlösung Kabel, Schnittstelle und Terminalprogramm: DM 130,—
!!! TPMON 64 ist das Terminalpgm f. Mailboxdialoge. Es kann ALLES. Briefe, Automatisch Parameter ... DM 69,—
!!! KOPIERSCHUTZ AUF SPUR 37 !! DM 50,—. Alle Angebote nur gegen VK oder NN bei M. Hoos, 2160 Stade, Hohenfriedberger 39

C64 Einführungsangebote C64
SIGNUM 64 DM 79,—
Entwerfen eigener Zeichensätze
35 KByte freier Basicspeicher
volle Drucker und Basicnutzung
eig. Visitenkarten, Briefköpfe, Etiketten, Firmenzeichen etc.
SUPERSPRITE 64 DM 59,—
Elektron. Zeichen von Sprites
voll kompat. zu allen Basicpr.
KOMPLETTPREIS nur DM 125,—
FIRMA H. GOEBEL SOFTWARE
Meißnerw. 11, 3501 Fuldabrück 1
Telefon: (0561) 581658

Geld zurück ??

C-64 o. CBM 3/4/8 vorhanden? Dann ermitteln Sie Ihre persönliche Steuererstattung mit dem komfort. Programm:

--- EST / LOJA 84 ---

Berechnet Einkommensteuer u. Lohnsteuer-Jahresausgleich. Auf neuest. Stand. Wahlweise Monitor- o. Druckerausgabe. Prgr. LSt-Tabelle 85 gratis!

2 Progr. auf Disk nur 38 DM!

Kostenlose Info mit Musterausdruck anfordern von Fa.:

Werner Eilers
Eichendorffstr. 3
6404 Neuhof

Deutsch-Lernprogramm für C64/Disk. 5 Lernbereiche, Datenpflege, Dateien, Begleitheft, ausbaufähig, Spitzengrafik, hoher Spielwert DM 40,— (Schein/Scheck). Info geg. Freiumschl. Bartens, Gehrenkamp 3a, 3220 Alfeld

C 64 ★ VC 20
★ Dop.reset C 64 + Floppy 1541 DM 36
★ Reset Userport DM 22
★ Einfacher Resettaster DM 10
★ HB SOFT, 35 Kassel, Pf 420222

Neu! Computer-Börse Wuppertal. Sie wollen Hard- oder Software kaufen/verkaufen? Adressenliste 4,— DM. Info kostenlos. R. Kohl, Bahnhofstr. 27, 56 Wuppertal 11, Tel. 0202/780809

Direktgekoppeltes Modem für C 64 mit aut. Wähleinrichtung und Teilnehmererkennung 300 Bd. Vollduplex Direktanschluß an C 64 galvanisch getrennt. Mit Grundsoftware für Schnittstelle und Wählautomatik Bausatz 287,90 DM fertig 317,90 DM. Die gültigen Postvorschriften sind zu beachten.
Linde Electronic, Pirolweg 34, 5190 Stolberg, 02402/24375

DATALIFE SS/DD 10 St. 59,— 100 St. 570,— Contr. Data SS/DD 10 St. 49,50 100 St. 475,— + 6,— Vers. Preisl. anford. W. Kotschenreuther, Gg-Buchner-Str. 29, 85 Nbg 10 (0911) 516739

Private Kleinanzeigen

!! TURBO-TAPE-MODUL C-64 NEU !!
10* schneller Load, Save, Verify, Merge, PRG bis zu 46 KB !!!!
Load + Save v. Maschinen-PRG (,1,1) + OLD, Mithörton im FS. DM 40,—
Kompatibel zu bek. Turbo-Tape.
RESET-Taster-Stecker..... DM 10,—
EPROM-Modul-Platine...... DM 25,—
AUTOSTART-Modul.......... DM 45,—
HCT P. Mertens, 02331/51807 ab 19 Uhr

FLOPPY 5MAL SCHNELLER + FEHLERKANAL + DIRECTORY **OHNE RAMVERLUST** MIT **TURBODISKMODUL;** SCHALTBAR; AUTOSTART; VOLLELECTRONISCH NUR 49 DM, DTO. ALS TURBOTAPEMODUL NUR 49 DM **VC20:4er MODULBOX** m. RESETT. 79 DM, KOPIERADAPTER; ADRESSADAPTER ..., **EPROMKARTEN** F. VC20 U. C64 ab 29 DM
HARD-SOFTW. R. WEISANG, BUTTERPFAD 14, 6682 OTTWEILER 4, TEL. 06858/556

Kopierschutz für C64/VC1541, Info gegen Freiumschlag von GEFDA, Pf. 2202, 6330 Wetzlar

DISKETTEN m. Garant. f. **Händler** weiße Ware ab **DM 3,35** 5¼" auch 96-tpi weiß von NASHUA und 8", Fa. 3A, Ringstr. 10, 8057 Eching, Tel.-Nr.: 08133/6116, Tx.: 527551

★★★★★★★★★★★★★★★★★
★★★★ **ACHTUNG !!!** ★★★★
★ **QUALITÄT ZU FAIREN PREISEN** ★
★ **USERPORT-STECKER** 7,50 ★
★ **USERPORT-RESET** m. Geh. 12,50 ★
★ **RESET-STECKER** 6-Pol. 4,50 ★
★ ab 3 St. 4,—; ab 50 St. 2,90 ★
★ Ihr Recorder als Speicher: ★
★ **KASSETTENINTERFACE** 29,50 ★
★ **JOYSTICKVERLÄNGERUNG:** ★
★ 2m 16,—; 3m 18,—; 5m 20,— ★
★ **STECKPLATZERWEITERUNG** ★
★ 5-fach umschaltbar m. Reset ★
★ als Bausatz komplett 68,00 ★
★ **SONDERANFERTIGUNG** mögl. ★
★ **VOLLES RÜCKGABERECHT** ★
★ Bei Vorauskasse Porto 1,90 ★
★ Postscheck 566373-204 / Hmb ★
★ NN mögl. **TEL 0451/72489** ★
★★ **KOPMANN & CZICHY** ★★
★★★ **COMPUTER-ZUBEHÖR** ★★★
★★★★ **Aegidienstr. 45** ★★★★
★★★★★ **2400 Lübeck** ★★★★★
★★★★★★★★★★★★★★★★★

Minispionkatalog 20 DM. Funk-Alarm-Katal. 20 DM. Donat, Pf. 420113 B, Köln

Alles für den 64'er
z.B. Epson RX-80 944,—
M-80 790,—
Floppy-Express 250,—
in Berlin, 3952920

+++ VC-64 +++ SPECTRUM +++
+ Für 2,— DM in Briefm. erhalten +
+ Sie meine neueste Softwareliste +
+ Lothar Hartmann, Postfach 82, +
++ 6535 Gau-Algesheim ++

Neu Sensationell Neu
QUICKLOAD-ACTIVATOR C64/VC1541 prägt Ihren bespielten Disketten in wenigen Sekunden eine bleibende Schnellade-Eigenschaft auf. Kein lästiges Vorladen eines Programms; keine Hardware-Änderung nötig! Aktivierte Disketten werden 6- bis 8mal schneller eingeladen, auch sofort nach Einschalten des Computers!
auf Diskette: **DM 59,—**
MACHO, Carl-Benz-Straße 30
7501 Marxzell 1, Tel. 07248/5395

MIDI - C 64
Midi-Interface für C 64, läuft mit allen gängigen Programmen, ready to use, 96,— DM N.N. inkl. MwSt. + Vers.-kosten
Musik-Elektronik-Butterweck
Hochstraße 75
4630 Bochum 6

Private Kleinanzeigen

★★★★★★★★★★★★★★★★★★
Qualitäts-DATENKASSETTEN
!!! **C-10 = 1,50 per/Stk** !!!
!!! **C-20 = 1,60 per/Stk** !!!
!!! **C-30 = 1,70 per/Stk** !!!
+ DM 3,50 Porto. Ab DM 30 frei!
Ausland + 4,50 Mengenrabatte: ab DM 75-5%; DM 100-8%; DM 150-12% Vorkasse, Nachnahme oder V-Scheck Anrufbeantworter, Tel. 040/6411981 E. Jensen; Fahrenkrön 49; 2 Hmb. 71
★★★★★★★★★★★★★★★★★★

COMMODORE 64
Spitzensoftware aus allen Bereichen. Günstig. Gratis-Info.
LOBIVIA-Soft, A. Eggli, Bahnhofstr. 10, CH-2544 Bettlach

★★★★★★★★★★★★★★★★★★
★ **preiswerte CBM 64-Hardware** ★
★ Markendisketten ab 4,50 ★ Eprom Modul 2 Steckpl. schaltbar 28,— ★ IEEE-Interface o. Speicherpl.-Bel. 259,— ★ Centronics-Kab. 68,— ★ Modem-Akust.-Kopp. kompl. 270,— ★ Prof. Doppelfloppy 2695,— ★ Prof. Eprommer 335,— ★ prof. Betriebssystem kompl. 129,— ★ Resettaster 9,50 ★ Disklocher ab 8,— (alle Preise inkl. MwSt.), Info gegen 2,— von J. Jeismann, Crispinstr. 4, 4600 Dortmund 50
★★★★★★★★★★★★★★★★★★

• C 64 Turbo Disk C 64 •
• Lädt 100 Blocks in 8 Sek. DM 25 •
• Turbo Tape II — Opt. und Akust. Kon. •
• zur Tonkopfjust. über TV DM 20,— •
• Bestell. 35 Kassel, Pf. 420222 •

SOFTWARE, DIE SPASS MACHT!
Commodore 64
SUMMER GAMES 2 Kass. DM 79,— **BEYOND CASTLE WOLFENSTEIN** Disk DM 129,—, **GHOST BUSTER** Disk DM 99,—, **BC'S QUEST FOR TIRES** Kass. DM 69,—, **STRUCTURED BASIC** dt. Mod. DM 189,—
VC 20
CHOPLIFTER Mod. DM 69,—
LODE RUNNER Mod. DM 69,—
Bei Nachnahme + DM 4,90. Katalog mit Qualitätssoftware für Arbeit und Unterhaltung gleich anfordern! **TELEDIENST**, Mainzer-Tor-Anlage 45v, 6360 Friedberg, Tel. 06031/91650, Btx 213 213

■■ **WIR KNACKEN DIE PREISE** ■■
■ C 64/20/ Hardware ■ P. inkl. MwSt. ■
■ USER-PORT-STECKER 8,90 ■
■ Turbo-Tape-Modul 10 x 59,90 ■
■ Joyst.: COMPETITION PRO 59,90 ■
■ Turbo-Disk-Modul 5,5 x 69,— ■
■ Spiel/Modulgehäuse 9,90 ■
■ Disk: SPRITE-EDITOR 39,50 ■
■ 24-Std.-Service: 0209/397862 ■
■ F. Brunken ■ Info: 2 DM ■ per NN. ■
■ Gladbecker Str. 123, 4650 Gelsenk. 2

C-64 + VC20 Flugtraining. Auch Versionen für 2001 bis 8032 lieferbar. Für VC20 +8 K (oder mehr) erforderl. Umfangreiche Auswertung Ihrer Flüge. Erklärung der Fluginstrumente. Steuerung mittels Tastatur oder Joystick.
A) Hubschraubersimulator. Hubschr. in Aktion. 9 Anzeigen im Cockpit, 3 Flugprogramme zur Wahl. 29 DM
B) Space Shuttle Landung. Echtzeitsimulation. 29 DM
C) Boeing 727-Simulator. Dieses Spitzenprogramm ist z. Anfänger- und Instrumentenflugschulung geeignet. Mit Anleitung 34 DM
Ab 2 Program. jedes Progr. minus 5 DM. Info gegen Rückporto.
Lieferung p. NN auf Kass. oder Disk. Fluging. F. Jahnke, Am Berge 1, 3344 Flöthe 1, Tel. 05341/91618

B&S — WIR HALTEN SIE AUF DEM LAUFENDEN — KOSTENLOSES INFO anfordern mit vielen Tips und interessanten Informationen, Comal zum Selbstkostenpreis gleich mit verlangen, Schreiben Sie — es lohnt sich
B&S — Waldstr. 23, 6209 Hohenstein 4

IGEL GEGEN HASE!!!
TURBO-TABE-MODUL für C-64: macht alle Kass.-operationen 10mal so schnell. Komp. zu Toolkits wie KFC-Super, VC-Extra, Mycom-One u.a. Belegt keinen BASIC-Speicherpl. Mit den zus. Befehlen 1. Old (Holt m. Reset o. New gel. Prgr. zurück). 2. Memory (Zeigt die Speicherbel. des C-64). 3. REM-Revers (Jede REM-Zeile wird rev. dargest.). Preis 45,— DM
TURBO-DISK-MODUL für C-64: macht Ihre Floppy ca. 7mal so schnell!!!! Belegt keinen v. Prgr. gen. Speicherplatz. N. Reset ern. Aufr. m. Sys. Durch Poke umschaltbar auf Drive 9. Drucker o. a. Ger. brauchen nicht ausgesch. werden. Arbeitet o. Einschr. mit allen Toolkits (z.B. Simon's BASIC) zus. Preis 59,— DM
!!! RESET-Taster nur 5 DM inkl. Porto!!!
EPROM-MODUL-KARTEN für alle Eprom-Typen lieferbar.
Computertechnik Ingo Klepsch
Tel. 02333/80202

★★★★★★★★★★★★★★★★★★
★★ NEU FÜR IHREN CBM 64 ★★
Machen Sie Ihren CBM 64 komfortabler mit einem neuen Betriebssystem. Viele bequeme Diskettenbefehle und viele neue, nützliche Funktionen durch ein neues Kernal-ROM. Keine Ladezeit! keine Speicherplatzbelegung! Ausf. Info geg. 1,— DM von J. Jeismann, Crispinstr. 4, 4600 Dortmund 50
★★★★★★★★★★★★★★★★★★

★★★ Telekommunikation ★★★
Akustikkoppler, Modems, Software, Mailboxsysteme. Rudolf Möllenbeck, Alte Linnerstr. 24, 415 Krefeld, Tel. 02151/20130 Mailb./801339

Qualitätszubehör für C64
Neu: Akustikk. S21D mit FTZ-Nr. 298,—
Datenm. 20 MHz, 9 Zoll, Grün 298,—
Typenradschreibmaschine Gabriele 8008 mit eingebautem IEC-Interface auch für CBM 4000/8000 998,—
Eprom-Platine für 4/8K (2732 oder 2764) 10,—
Eprom-Platine für 16/32 K (4 2732 oder 4 2764) 59,—
Eprom-Brenner für 2716-27256, 2532, 2564, EEPROM 2816 348,—
IEEE 488-Interface mit zwei 8 K-Epromsockel 298,—
Centronics-Interface, belegt keinen Speicherplatz 248,—
Experimentierkarte mit 2 CIA 6526A, 600 Lochrasterpunkte Durchkontaktiert, zwei 8 K-Epromsteckplätze 298,—
Bausätze und Leerplatinen auf Anfrage! Ausführliche Informationen gegen Freiumschlag.
Computerelectronic Bernd Luda Zollnerstr. 31, 8600 Bamberg Telefon 0951/36839

★★★ **Staubschutzhauben** ★★★
C 64/VC 20/VC 1541 je DM 15,—, RX 80/MPS801 je 25,—. Ab 3 St. 10% Rab. Scheck/Schein. NN + DM 2,— Gebühr. Fa. Weber, Pf. 201237, 56 Wuppertal 2

★★★ C 64 ★ VC 20 ★★★
★ Turbolüfter vermeid. Wärmestau in Floppy 1541, verhind. Dejustierung des Schreib-Lese-Kopfes DM 78,— HB SOFT, 35 Kassel, Pf 420222

Disketten 5¼-Zoll-Versand, Maxell MD 1, 10 St. 65,—
Scotch 744 D, 10 St. 64,— Info 2,— Richter, 4047 Dormagen, Postfach 100527

DEK DEK DEK DEK DEK DEK DEK DEK
Commodore 64 Zubehör:
80-Zeichen-Karte, Simon's-, Exbasic + CP/M-kompatibel DM 298,—
IEEE-488-Bus inkl. DOS 5.1 DM 249,—
Eprommer für 2758-27256 DM 257,—
128 KB-Epromkarte + Softsw. DM 189,—
64 KB-Epromk. + Res.Taste DM 89,—
16 KB-Epromk. + Res.Taste DM 59,—
Steckplatzerw. 2 Steckpl. DM 59,—
2732: DM 20 2764: DM 25
27128: DM 60,—
Eprom-Brennservice DM 5/Stck.
CBM-Reparaturservice
Liste anfordern! Tel. 06658/615
DEK Dassow Elektronik KG
Rauschelbach 7, 6416 Poppenhausen

NEU! NEU! pp-teachware NEU! NEU!
Software für den Schul- und Bildungsbereich von Pädagogen, z. B.
Deutsch, Mathe, Sprachen, Physik u.a.
Preisliste anfordern bei:
pp-teachware G. Lowak & W. Euerle
Schloßstraße 7, 5400 Koblenz
NEU! NEU! pp-teachware NEU! NEU!

★★ **Recorderinterface C64/VC 20** ★★
zum Anschluß eines handelsüblichen Recorder o. Tonb. als Datenspeicher. Fertiggerät 33,— DM mit Kabel. ★ Userport-stecker — 7,90 DM ★ Kabel-Userport-Userport zum verbinden von zwei Commodore-Comp. m. Programm 3 m lg. — 39,— DM. Fa. BOI / Hoye 1 / 2225 Schafstedt, T. 04805/380

★★★★★ **CBM-64** ★★★★★
★ Programmierbare ★
★ Steckdose .... DM 160,00 ★
★ Einsteiger-Disk.... DM 28,50 ★
★ User-Port-Tester... DM 24,50 ★
★ und vieles mehr. Info 64r bei ★
★ J. Hofer ★ Hermine-Maierheuser-Str. 7 ★ 7515 Linkenheim-Ho. ★
★★★★★★★★★★★★★★★★★★

64 ★ 64 ★ 64 ★ 64 ★ 64 ★ 64 ★ 64 ★
EPROM-MODUL-GENERATOR macht aus Ihren Basic- oder Maschinencodeprogrammen brennfertige Eprommodule. Mit Autostart, STOP-Schutz. DISK 79 DM
Switch 2 die Super Epromkarte. Kann nach Programmstart softwaremäßig ausgeschaltet werden (Speicherplatz ist wieder frei). Reaktivierung nur mit Hardwarereset möglich.
Für Eproms der 27..er Serie. 39 DM
TURBO-TAPE-MODUL macht Ihrer Datasette Beine. LOAD, SAVE und VERIFY mit der Geschwindigkeit eines Floppies.
Im Gehäuse mit Hardwarereset 69 DM
TURBO-DISK-MODUL LOAD von Diskette 6mal schneller.
Im Gehäuse mit Hardwarereset 69 DM
Bestellen Sie noch heute unser kostenloses Info!
KELLERMANN Computer-Shop, Konradin-Adenauer-Platz 30, 4030 Ratingen 4, Tel. 02102/33918

C=64: Eprom-Platine 2x 4 KB DM 40, 4x 4 KB DM 59, dito mit VC-SUPER (64'er 5/84) DM 198, Steckplatzerw. 4fach DM 148, Epsondruckerinterf., CBM Grafik DM 268, Eprommer bis 2732 DM 175, dito bis 16 KB = 27128 DM 245, Vizawrite Disk/Modul ab 298, Vizastar DM 398, Userportkabel DM 65, Epsondrucker mit Grafikinterf. ab 1299, CP-80 F/T Drucker DM 928, dito mit Interf. DM 1150, Druckerpuffer, Farbmonitore (TAXAN) ab DM 848 uva., Hartge-C., Fohlenweide 16, 6836 Oftersheim, 06202/52675

JANN DATENTECHNIK

Sie suchen für Ihren

C 64 ein IEEE-488-(IEC-)Interface

das wirklich professionell funktioniert?
Hier ist es:

- Vollständig steckbar
- Datenübertragung mit voller IEEE-488-Geschwindigkeit
- Durch neues Betriebssystem (kompatibel zum Original-64er) kein Speicherplatzbedarf und wirklich kompatibel zu **allen** Programmen (auch Maschinenprogramme mit Direkteinsprüngen in das Betriebssystem)
- Modulspeicherbereich ($8000-$9FFF) sowie $C000-$CFFF können weiterhin benutzt werden
- DOS 5.1 voll implementiert (erweiterte Funktionen)
- Serieller C 64-Bus sowie V.24 (RS232) bleiben ohne Umschaltung weiter verwendbar
- Kompatibel zur EX 80/C64

IEEE-488/64 betriebsfertig aufgebaut ..... **DM 249,—**
passendes Anschlußkabel hierfür,
1,50 m/Flachband ..... **DM 49,—**

Glimmerweg 22, 1000 BERLIN 47, Tel. (0 30) 73 11 84

JANN DATENTECHNIK

MYCOM ONE

... sicherlich eine der besten derzeit erhältlichen Befehlserweiterungen für Ihren C 64.

- Implementiertes FAST-TAPE (10mal schneller)
- Implementierter Maschinensprache-Monitor
- Softwaremäßig vollständig abschaltbar (belegt dann keinerlei RAM-Platz)
- Befehlsübersicht: AUTO / RENUM / DEL / TRACE / FIND / DUMP / HELP / OLD / CAT / STATUS / DLOAD / GAME / DVERIFY / DMERGE / DSAVE / DISK / TC / PUT / GET / COMP / MERGE / PAUSE / REPEAR / ESC / OFF / MON / FREE / TOKEN / SWITCH / KILL / LLIST / LPRINT / ELIST / EPRINT / CLIST
- MYCOM-ONE-Steckmodul mit ausf. deutscher Bedienungsanleitung ..... **DM 139,—**
- MYCOM TWO — wie Mycom one, jedoch statt Maschinensprache-Monitor: FASTLOAD v. Diskette (7x schneller), FASTCOPY (3 Min.), HARDCOPY ..... **DM 139,—**

Glimmerweg 22, 1000 BERLIN 47, Tel. (0 30) 73 11 84

JANN DATENTECHNIK

Jetzt auch für Ihren C 64:

Unsere 80-Zeichen-Karte für professionelle Datenverarbeitung

- Vollständig steckbar — BAS-Ausgang
- Kompatibel zu Exbasic, Simons-Basic, Austrocomp, Austrospeed, zur CP/M-Karte u.v.m.
- Sämtliche Bildschirm-Editierfunktionen des CBM 8032 vorhanden (Bildschirmfenster-Definition, Zeileneditierung, Ton, Auf- und Abwärts-Scrolling usw.)
- Zeichendarstellung mit 8x8-Punkt-Matrix
- 40 + 80 Zeichen gleichzeitig verwendbar und softwaremäßig umschaltbar
- Video-RAM direkt durch PEEK und POKE beschreibbar
- Nachweis erstklassiger Textverarbeitungsprogramme auf Anfrage

EX 80/64 funktionsfertig aufgebaut, inkl. ausführlicher Bedienungsanleitung sowie Zeichengeneratorprogramm für die Erstellung zusätzlicher Zeichensätze (damit die Umlaute dort liegen, wo Ihr Drucker sie druckt) ..... **DM 298,—**

Glimmerweg 22, 1000 BERLIN 47, Tel. (0 30) 73 11 84

Müllabfuhr im Comp
Die Garbage Co

Wer viel mit Strings arbeitet, wurde wohl schon öfter mit einem Problem konfrontiert, der Garbage Collection. Um was es dabei geht, und warum es sich lohnt, sich etwas mehr mit Strings zu beschäftigen, erfahren Sie in folgendem Artikel. Dies soll zugleich der Auftakt zu einer Serie sein, in der wir uns mit speziellen Problemen und Programmiertechniken beschäftigen werden.

Kennen Sie das? Sie haben ein Basic-Programm geschrieben, starten es, und eine Zeit lang läuft es perfekt. Und plötzlich, wenn Sie nur mal zwei Strings vertauschen oder die FRE-Funktion aufrufen, spielt Ihr Computer nicht mehr mit, ja, er reagiert nicht einmal mehr auf die STOP-Taste. Warten Sie dann einige Sekunden oder Minuten, ist der Spuk vorbei. Ihr Computer tut so, als sei nichts geschehen. In solchen Augenblicken hat der Basic-Interpreter mal wieder Müllabfuhr gespielt, er hat die »Garbage Collection« durchgeführt. Dieses Wort ist der Alptraum mancher Programmierer, die schnelle Programme schreiben möchten, welche Strings in großen Mengen benutzen.

Hier sind schon einige Stichworte gefallen: Bei der Garbage Collection handelt es sich wohl um etwas, das mit Strings zu tun hat und wohl auch mit Speicherplatz.

Um allerdings genau zu verstehen, was es mit dieser Routine auf sich hat, und welche programmtechnischen Wege man gehen muß, um sie zu meiden, müssen wir uns erst einmal mit dem computerinternen Aufbau von Strings und Stringarrays beschäftigen.

Was ist ein String?

Ein String ist eine Zeichenkette, also eine Aneinanderreihung von Zeichen aus dem Zeichensatz eines Computers. Unserem Computer zeigen wir an, daß er die nächste Eingabe als String betrachten soll, indem wir selbigem ein Gänsefüßchen voranstellen. Mit einem solchen geben wir ihm auch das Ende eines Strings bekannt.

Ein String hat immer eine bestimmte Länge, sie entspricht der Anzahl der in ihm enthaltenen Zeichen. Die Länge darf zwischen 0 und 255 betragen, weil der Computer nur über Umwege mehr als 255 Zeichen auf einmal überblicken könnte. Bei einer Länge von Null spricht man übrigens auch von einem Leerstring, weil keine Zeichen in ihm enthalten sind, nicht zu verwechseln mit einem String, der Leerzeichen (Spaces, Blanks) enthält.

Da es unpraktisch wäre, jedesmal, wenn ich einen bestimmten String bearbeiten will, diesen vollständig einzugeben, gibt es die Stringvariablen. Eine Stringvariable ist nichts anderes als ein Platzhalter für einen String. Wenn ich beispielsweise der Stringvariablen A$ den String »BORIS SCHNEIDER« zuteilen möchte, so schreibe ich:

A$="BORIS SCHNEIDER"

Das Dollarzeichen hinter dem Variablennamen zeigt dem Computer an, daß es sich hier um eine Stringvariable handelt. Nun kann ich im weiteren Verlauf jedesmal, wenn ich »BORIS SCHNEIDER« eingeben müßte, A$ dafür schreiben. Allerdings können Basic-Befehle so nicht abgekürzt werden. Wenn ich eingebe:

A$="PRINT":A$ 3*5

so erhalte ich nicht das gewünschte Ergebnis, 15, sondern schlicht und einfach SYNTAX ERROR. Das liegt daran, daß Basic-Befehle intern anders codiert sind als Strings.

Aufbau einer Stringvariablen

Wie sieht nun eine Stringvariable aus? Dazu betrachten wir ein Speichermodell des Computers. Die folgenden Erläuterungen beziehen sich auf Bild 1.

Beim Ablauf eines Programms läßt sich der Speicher in folgende fünf Bereiche aufteilen, deren Grenzen allerdings flexibel sind und sich laufend ändern:

— Basic-Programm

Das gerade im Speicher befindliche Programm liegt ganz unten, angefangen bei den kleinsten erreichbaren Adressen.

— Variable

In diesem Bereich befinden sich alle Variablen, Zahlvariablen wie auch Stringvariablen. In den Stringvariablen ist allerdings nicht der String selbst enthalten, sondern nur ein Zeiger auf die Adresse, wo sich der eigentliche String befindet.

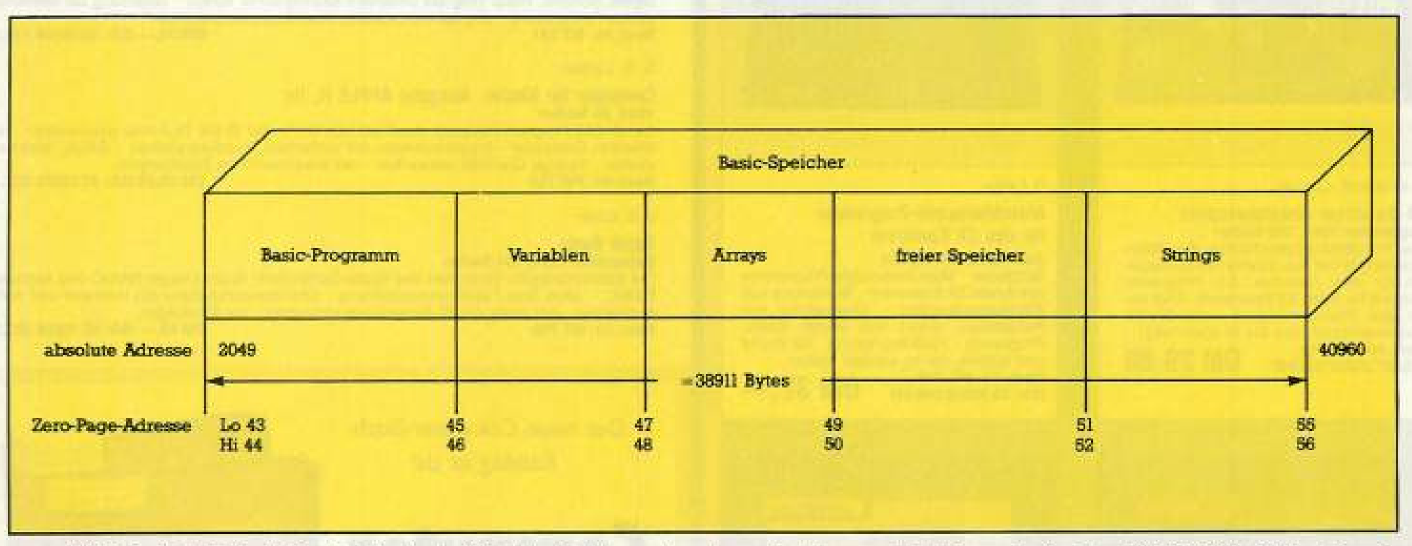
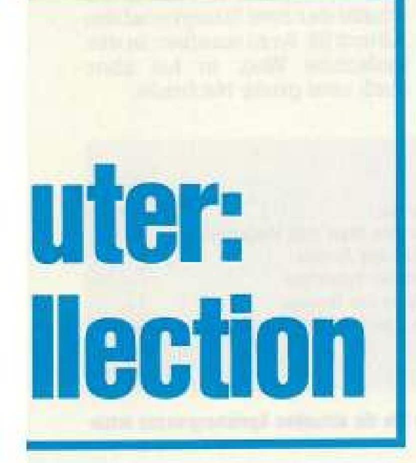


Bild 1. So ist der Basic-Speicher im Prinzip aufgeteilt. Bekannt sind normalerweise nur die Anfangs- und Endadresse des Speichers (2049/40960). Mit den Zeigern in der Zero-Page können alle Adressen abgefragt werden. Die Basic-Anfangsadresse errechnet sich zum Beispiel aus PRINT PEEK(43)+PEEE K(44)x256, der Variablenanfang aus PRINT PEEK(45)+PEEK(46)x256 und so weiter.



einer Ausnahme, aber davon später).

Nun kommen wir aber wieder auf die Stringvariablen selbst zurück.

Wie schon erwähnt, steht, speichertechnisch gesehen, in einer Stringvariablen gar kein String, sondern nur die Adresse, wo wir den Inhalt der Stringvariablen finden können. Das bemerken wir allerdings normalerweise nicht, da der Basic-Interpreter für uns automatisch den String an der entsprechenden Adresse abholt.

Der genaue Aufbau einer Stringvariablen steht in Tabelle 1. Hier einige Erläuterungen dazu:

Jede Variable ist durch eine Zweizeichen-Kombination gekennzeichnet. Das erste Zeichen muß ein Buchstabe, das zweite Zeichen darf auch eine Ziffer sein. Durch das Anhängen eines »$« definieren Sie die vorstehende Variable als Stringvariable. Intern speichert der Computer nun das Dollarzeichen nicht mit, sondern kennzeichnet eine Stringvariable, indem er zum Code des zweiten Buchstabens 128 addiert. So werden nur zwei Speicherstellen benötigt, um den Variablennamen zu speichern. Bei Zahlvariablen wird nichts addiert, bei Integervariablen (Ganzzahl) zu beiden Codes jeweils 128.

Das dritte Byte gibt die Länge des Strings an. Dies ist notwendig, da die eigentlichen Strings in ihrem Speicherbereich ohne Trennzeichen einfach aneinandergehängt sind. Wüßte der Computer nicht, wie lang der gefragte String ist, so könnte er auch nicht sein Ende feststellen. Da kein Trennzeichen verwendet wird, darf in einem String auch jedes Zeichen, das der Computer kennt, vorkommen. Außerdem wird so der Speicherplatz am besten ausgenutzt.

Die Bytes 4 und 5 enthalten die Adresse des Strings im Low/High-Byte-Format. Um diese beiden Bytes in die entsprechende Zahl umzuwandeln, müßte folgende Rechnung durchgeführt werden:
Byte 4 + (256 x Byte 5)

Die Bytes 6 und 7 enthalten beide den Wert Null. Wenn Sie nach dem Sinn und Zweck fragen, so ist die Antwort, daß es für den Computer einfacher ist, wenn alle Variablentypen, Zahlen wie Strings, die gleiche Länge haben. Eine Zahlvariable belegt nun genau 7 Bytes, deswegen werden Stringvariable auf diese Länge aufgefüllt.

Der Teil der Stringvariablen, der Länge und Adresse angibt, wird als Descriptor bezeichnet.

```
0 REM +++ STRINGVERTAUSCHER +++
1 REM  GESCHRIEBEN AM 07.10.84
2 REM     VON BORIS SCHNEIDER
3 REM
4 REM SYNTAX:
5 REM SYS STARTADRESSE(STRING1,STRING2)
6 REM
7 REM BELIEBIG IM SPEICHER VERSCHIEBBAR
8 :
10 DATA  32,250,174, 32,158,173, 32,143
20 DATA 173,165,100,133,247,165,101,133
30 DATA 248, 32,253,174, 32,158,173, 32
40 DATA 143,173,160,  0,177,247,133,249
50 DATA 177,100,145,247,165,249,145,100
60 DATA 200,192,  3,208,239, 32,247,174
70 DATA  96,  0,  0,  0,  0,  0,  0, 0
80 :
100 INPUT"STARTADRESSE";SA
110 FOR I = SA TO SA+48
120 :READ X:POKEI,X:CS=CS+X
130 NEXTI
140 IF CS<>7314THEN PRINT"FEHLER!!"
150 END
READY.
```

Listing 1. Mit diesem Programm können Sie Strings vertauschen, ohne daß dabei Stringmüll entsteht.

— Arrays
Hier befinden sich die Arrays, auch Felder genannt. Wenn Sie mehr als elf Elemente enthalten sollen, müssen Sie vor ihrer Benutzung erst durch den DIM-Befehl definiert werden, damit der Basic-Interpreter genügend Speicherplatz in diesem Bereich bereitstellt.

— Freier Speicher
Freier Speicher existiert interessanterweise nicht am oberen oder unteren Ende des Speichers, sondern in der Mitte, zwischen Arrays und Strings. Das hat aber enorme Vorteile, wie wir noch sehen werden.

— Strings
Hier sind sie endlich, am oberen Ende des Speichers, die von uns gesuchten Strings. In diesem Bereich befinden sich, dicht aneinander gepackt, die Inhalte der Stringvariablen (mit

Stringarrays

Nun gibt es aber neben den normalen Stringvariablen auch Stringarrays. Dies sind ein- oder mehrdimensionale Platzhalter. Am besten läßt sich das mit einem Beispiel erklären. Mit dem Befehl DIM A$(100,5) wird ein zweidimensionales Stringarray definiert. In diesem Array lassen sich bestimmte Strings mit Hilfe von zwei Koordinaten ablegen, zurückholen oder bearbeiten. Das soeben definierte Array läßt sich mit genau 606 Strings auffüllen. Man darf die erste Koordinate, oder besser gesagt den ersten Index, von 0 bis 100 und den zweiten Index von 0 bis 5 angeben. Dies sind 101 x 6 = 606 Platzhalter für Strings. Der Vorteil von Stringarrays ist, daß mit ihnen berechnete Zugriffe auf Strings möglich werden. So lassen sich zum Beispiel Strings in eine Reihenfolge bringen, diese Reihenfolge aber auch beliebig ändern, indem zwei Strings miteinander vertauscht werden. Anders gesagt sind zum Beispiel Sortierprogramme erst mit Stringarrays möglich geworden. Denn es gibt wohl keinen Weg, ohne große Tricks und viel Speicheraufwand die Inhalte der Stringvariablen A$, B$ .... Z$ zu sortieren. Mit einem Array ist dies sehr einfach möglich. Doch nun zu den Stringarrays selber.

Auch hierzu habe ich wieder eine Tabelle zusammengestellt (Tabelle 2).

Wie Sie schon auf einen Blick sehen, ist der Aufbau von Stringarrays ungleich komplizierter als der von normalen Stringvariablen.

Hier bezeichnet man den gesamten Block der einzelnen Stringdescriptoren als Arraydescriptor (description = Beschreibung, Darstellung).

In den ersten beiden Bytes steht, wie bei normalen Stringvariablen, der Name im Interpretercode. Auch hier wird zum zweiten Byte 128 addiert. Als nächstes wird, wieder im Low/High-Format, die Gesamtlänge des Arrays angegeben (Byte 3 und 4). Diese Gesamtlänge umfaßt sowohl den Arraydescriptor, wie alle nachfolgenden Stringdescriptoren.

Im Byte 5 wird die Anzahl der Dimensionen angegeben. Im Beispiel A$(100,5) wäre dies 2, da ja zwei voneinander abhängige Indizes vorhanden sind. Man spricht auch von einem zweidimensionalen Array oder Feld. Jeder Index definiert also eine Dimension. Da ein Byte maximal den Wert 255 aufnehmen kann, wäre dies auch die höchstmögliche Dimensionierung.

Nun steht hintereinander die Anzahl der Elemente jeder einzelnen Dimension in absteigender Reihenfolge. Diese Angaben sind wieder im Low/High-Format. Um auf unser Beispiel zurückzukommen: Zuerst würde hier eine 5 stehen und dann eine 100. (Beide im Low/High-Format!)

Damit wäre dann der Arraydescriptor abgeschlossen; an ihn schließen sich die Descriptoren der einzelnen Strings an. Jeder Stringdescriptor belegt 3 Bytes: Das erste gibt die Stringlänge, die beiden anderen die Speicheradresse des Stringinhalts an. Diese Stringdescriptoren stehen in folgender Reihenfolge hinter dem Arraydescriptor: Zuerst wird der Index der höchsten Dimension hochgezählt, dann der der zweithöchsten und so weiter, bis zur Dimension 1. Auch hier greife ich kurz auf unser Beispiel zurück: Der erste angegebene Stringdescriptor bezieht sich auf den String A$(0,0), der zweite auf A$(0,1), der sechste auf A$(0,5) der siebte auf A$(1,0), ... der 606te auf A$(100,5). Damit wäre dann auch die Liste der Stringdescriptoren zu Ende.

Diesen kleinen Ausflug in die Speichertechnik möchte ich mit einer Formel beenden, mit der Sie die Größe eines Arraydescriptors und seiner Stringdescriptoren berechnen können. Sie lautet: 5 + (Anzahl der Dimensionen) x 2 + (Anzahl der Elemente Dimension 1) x (Anzahl Elemente Dimension 2) x ... (Anzahl Elemente Dimension n) x 3.

Wichtig wäre hierbei, daß Sie bei der Angabe der Anzahl der Elemente beachten, daß die Zählung immer bei Null beginnt. A$(100,5) belegt also 5 + 2 x 2 + 101 x 6 x 3 = 1827 Bytes, die In-

halte der einzelnen Variablen natürlich ausgeschlossen.

Ein Blick in den Speicher

Nun wissen wir also, wie Stringvariablen und -arrays aussehen. Was passiert nun aber, wenn ich einen String anlege oder ihn bearbeite?

Um uns die Stringverarbeitung des Computers genauer anzusehen, bedienen wir uns ein paar speichertechnischer Tricks.

Wie schon erwähnt, stimmen die Adressenangaben, die ich bei unserer Beispielspeicherbelegung gemacht habe, nicht. Der Beginn eines Basic-Programms liegt normalerweise an der Adresse 2048, das Ende des Speichers bei 40959.

Die Grenzen zwischen den einzelnen Bereichen sind, wie gesagt, nicht exakt festlegbar, da sie sich laufend ändern. Sollten Sie sich für die gerade bestehenden Grenzen interessieren, so können Sie sie mittels PEEK erfragen. In den in Tabelle 3 angegebenen Speicherstellen stehen die jeweiligen Grenzen im schon erwähnten Low/High-Format. Um also beipielsweise den Beginn der Strings auszudrucken, müssen Sie folgendes eingeben:

PRINT PEEK(51)+256xPEEK(52)

Wir beschreiten nun einen sehr ungewöhnlichen, aber wirkungsvollen Weg. Wir verkleinern unseren Speicher so, daß er auf den Bildschirm paßt, und legen den Bildschirmspeicher an diese Stelle. Wir können dann zwar nicht mehr sehen, was wir eintippen, haben aber einen vollen Überblick über den Speicher.

Geben Sie bitte folgendes ein:
FORI=2060TO3072:POKEI,32:NEXTI, POKE 56,11:POKE 53272,37:CLR.

Sie haben im Augenblick zirka 750 Bytes freien Speicherplatz, der Rest wurde vorhin für Basic gesperrt.

Tippen Sie nun mal (blind) A$="ihr name". Sie sehen an zwei Stellen auf dem Schirm ein paar Grafikzeichen. Drücken Sie gleichzeitig die SHIFT- und die COMMODORE-Taste, um auf Kleinschrift umzuschalten. Nun erkennen Sie Ihren Namen, der im String A$ gespeichert wurde und an der obersten Speichergrenze steht.

Die Stringvariable selbst am oberen Bildschirmrand können Sie auch jetzt nicht entziffern, da sie im Interpretercode gespeichert wird. Zumindest können Sie aber die Stringinhalte lesen.

Jetzt wird es interessant: Tippen Sie mal A$="BORIS SCHNEIDER".

Sie werden bemerken, daß ihr Name stehengeblieben ist, während mein Name einfach vorne angehängt wurde.

Dies ist der Haken der Stringverarbeitung, denn Ihr Name ist jetzt, bitte nehmen Sie es nicht persönlich, zu Stringmüll geworden. Er steht immer noch im Speicher herum, obwohl kein Stringdescriptor mehr auf ihn zeigt.

Dies macht einen großen Teil der gegenüber anderen Computern hohen Geschwindigkeit der Stringverarbeitung aus, denn neue Strings werden einfach vor die anderen gesetzt, ohne daß diese überprüft werden, ob sie noch gültig sind. Hier findet sich dann auch die Erklärung, warum freier Speicher gerade in der Mitte anzutreffen ist, zwischen Arrays und Strings. Denn während die Arrays und anderen Variablen mit zunehmender Zahl nach oben wachsen, wachsen die Strings nach unten. Der dazwischenliegende Speicher wird immer kleiner.

Byte	Bedeutung
1	Buchstabe 1 des Stringnamens im Interpretercode
2	Buchstabe 2 des Stringsnamens im Interpretercode + 128
3	Länge des Strings
4	LSB der Adresse des Strings
5	MSB der Adresse des Strings
6 & 7	Auffüllbytes, immer =0

Tabelle 1. Der Aufbau einer normalen Stringvariablen.

Byte	Bedeutung
1	Buchstabe 1 des Arraynamens im Interpretercode
2	Buchstabe 2 des Arraynamens im Interpretercode + 128
3	LSB der Länge des Arrays
4	MSB der Länge des Arrays
5	Anzahl der Dimensionen
6 & 7	LSB & MSB der Anzahl der Elemente in der höchsten Dimension =n
8 & 9	LSB & MSB der Anzahl der Elemente der Dimension n-1
...	u.s.w. bis:
? & ?	LSB & MSB der Anzahl der Elemente der Dimension 1
Ende	Jeweils drei Byte Stingdescriptoren, sortiert in aufsteigender Reihenfolge, wobei der letzte Index als erster durchläuft.

Tabelle 2. So sehen Stringarrays im Computerspeicher aus. Nähere Erläuterungen im Text.

Adresse	Bedeutung
43,44	Start des Basic-Programms
45,46	Ende des Basic-Programms, Start der Variablen
47,48	Ende der Variablen, Start der Arrays
49,50	Ende der Arrays, Start freier Speicher
51,52	Ende freier Speicher, Start der Strings
55,56	Ende der Strings, Ende des für Basic verfügbaren Speichers.

Tabelle 3 enthält die Adressen, unter denen Sie die aktuellen Speichergrenzen erfahren können.

Und was passiert nun, wenn Strings und Arrays »zusammenstoßen«?

Hier gibt es zwei Möglichkeiten: Da ja kein Speicherplatz mehr frei ist, wird OUT OF MEMORY angezeigt, oder ...

Wie Sie sich vielleicht schon gedacht haben, können sich unter den Strings eine Vielzahl von Müllstrings befinden, die einfach nicht mehr gebraucht werden, sondern nur Speicherplatz schlucken. Könnte man diese entfernen, und nur noch die tatsächlich verwendeten Strings übriglassen, so wäre wieder eine Menge Speicher für weitere Anwendungen frei.

Nun sind wir also endlich beim Stichwort: Garbage Collection. Dies ist die vorhin angesprochene Möglichkeit der Müllbeseitigung.

Problemkind: Das Vertauschen

Bei der Garbage Collection wird sämtlicher vorhandener Stringmüll entfernt; so wird der größtmögliche Speicherplatz frei.

Wie arbeitet nun die Garbage Collection? Auch dies zeigt man am besten an einem Beispiel. Am praktischsten finde ich das Stringvertauschen. Sie wissen ja wahrscheinlich, wie man normalerweise zwei Strings vertauscht:
H$=A$: A$=B$: B$=H$

Diese Sequenz vertauscht die Inhalte der zwei Stringvariablen A$ und B$. So zu tauschen ist der einfachste Weg, er hat aber auch zwei große Nachteile:

— Es wird eine zusätzliche Hilfsvariable, hier H$ benötigt.
— Es entsteht eine Menge Stringmüll, nämlich 150 Prozent!

Schauen wir uns mal genau an, was bei der oben genannten Sequenz im Speicher passiert.

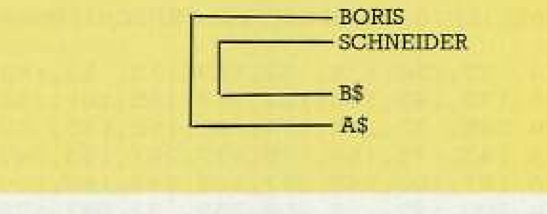


Bild 2. So sieht der Speicher nach dem Befehl A$="BORIS":B$="SCHNEIDER" aus. Oben stehen die Strings, unten die dazugehörigen Descriptoren.

Wenn Sie es selbst am Bildschirm verfolgen wollen, geben Sie bitte nochmals die oben genannten POKE-Befehle ein. Ansonsten betrachten Sie bitte Bild 2 bis 6.

Zuerst definieren wir die zwei Strings A$ und B$:
A$="BORIS" : B$="SCHNEIDER" (Sie können selbstverständlich jeden anderen Stringinhalt wählen.) Sie sehen, daß das Betriebssystem zwei Strings im Speicher abgelegt hat; die Descriptoren der entsprechenden Variablen zeigen auf diese (Bild 2).

Nun folgt der erste Vertauschungsschritt: H$=A$. Sie sehen, daß nun zweimal der String "BORIS" im Speicher steht, einmal als Inhalt von A$, einmal von H$ (Bild 3).

Der nächste Schritt wäre: A$=B$.

Schon wieder wird ein schon vorhandener Inhalt nochmals angehängt, in diesem Falle das »SCHNEIDER«. Das erste »BORIS« ist jetzt schon Stringmüll, da kein Descriptor auf ihn zeigt (Bild 4).

Als letztes käme noch: B$=H$. Nochmals wird der String »BORIS« in den Speicher geschrieben. Damit ist auch der vorheri-

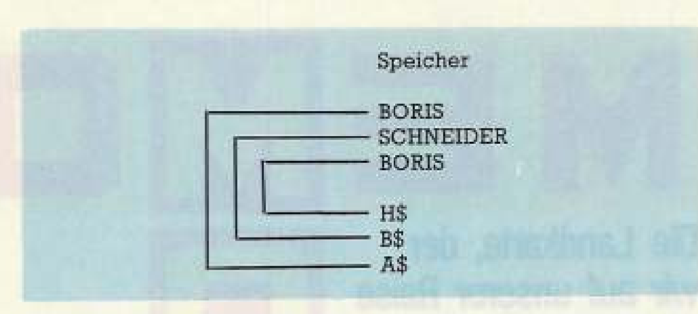


Bild 3. H$=A$. Bitte beachten Sie, daß in Wirklichkeit die Strings nicht unter- sondern in »Wirklichkeit« nebeneinander stehen.

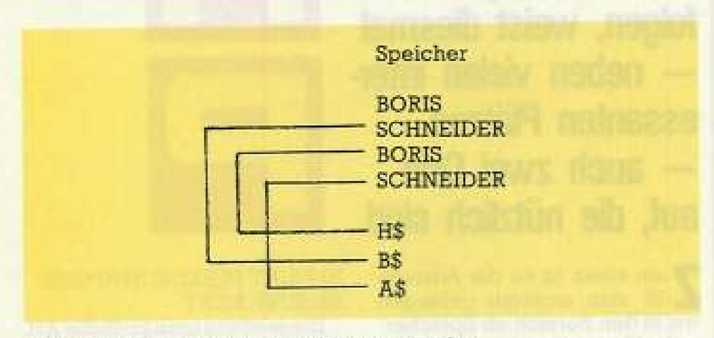


Bild 4. Nach A$=B$ ist der erste Müllstring entstanden

ge Inhalt von B$, der erste »SCHNEIDER«, zu Stringmüll geworden (Bild 5).

Zum Schluß wäre zu beachten, daß wir ja nun die Variable H$ nicht mehr benötigen, da sie nur Zwischenspeicher bei der Vertauschung war. Wenn wir also schreiben: H$="", wird auch das zweite "BORIS" zu Stringmüll (Bild 6).

Nun rechnen wir mal: Fünf Strings stehen im Stringspeicher, aber nur zwei werden benötigt. Eine traurige Bilanz, wenn man bedenkt, daß wir ja nur zwei Stringvariablen vertauscht haben. Denn bei jeder Stringfunktion entsteht Stringmüll. Probieren Sie doch mal A$=A$, und Sie werden sehen, wie schnell sich Stringmüll entwickeln kann.

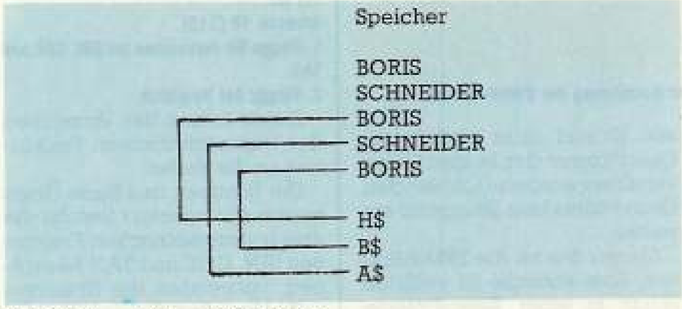


Bild 5. Und so sieht's nach B$=H$ aus

Allerdings gibt es eine löbliche Ausnahme, die man immer ausnutzen sollte. Wenn Sie Stringvariablen in einem Programm definieren, zum Beispiel: 10 A$="HALLIHALLO", so werden Sie am oberen Speicherende vergeblich nach "HALLIHALLO" suchen! Denn im Programmtext ist dieser String ja schon enthalten, wieso sollte der entsprechende Descriptor nicht einfach auf die entsprechende Stelle im Programmtext zeigen? Das tut er auch; zu beachten wäre aber, daß bei jeder weiteren Manipulation an dieser Stringvariable ein String im Stringspeicher abgelegt wird; es können also beim Vertauschen mehrerer im Programm definierter Stringvariablen ohne weiteres wieder Müllstrings entstehen.

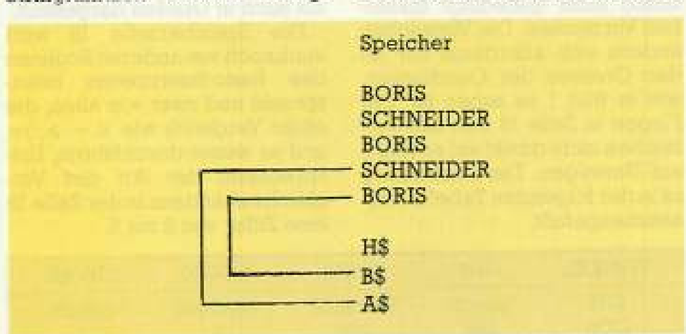


Bild 6. Das ganze entwirrt sich bei H$=""

Ich würde Ihnen empfehlen, einige andere Stringoperationen selbst auszuführen und sich die Effekte mit Hilfe des oben genannten Tricks selbst anzusehen.

Doch nun endlich zurück zur Garbage Collection und unserem Beispiel. Wir können den entstandenen Stringmüll durch Aufrufen der FRE-Funktion beseitigen, zum Beispiel: PRINT FRE(0).

Garbage Collection

Wenn Sie dies bei unserem Beispiel machen, dann sehen Sie, wie blitzschnell der Speicher umorganisiert wird; leider zu schnell, denn wir bekommen gar nicht mit, was passiert. Deswegen gehen wir also die Verfahrensweise der Garbage Collection an unserem Beispiel durch (Bild 7 und 8).

Die Garbage Collection durchsucht den gesamten Variablenspeicher nach dem String, der den am höchsten liegenden Pointer hat. Dabei werden alle Strings, die die Länge Null haben, übergangen, da sie ja keinen Inhalt haben können.

In unserem Fall ist das ziemlich einfach, da wir ja nur zwei richtige Strings haben, A$ und B$. H$ hat ja die Länge 0, wird also nicht beachtet.

Der String, der den höchsten Descriptor hat, ist bei uns A$, denn der Inhalt von A$ steht an einer höheren Stelle als B$. Daraus folgt aber auch sofort, daß alles, was über dem Inhalt von A$ steht, Stringmüll sein muß, da ja kein Descriptor mehr darauf weist. Also kann der Inhalt von A$ an die obere Speichergrenze geschoben, und der Descriptor von A$ entsprechend geändert werden (Bild 7).

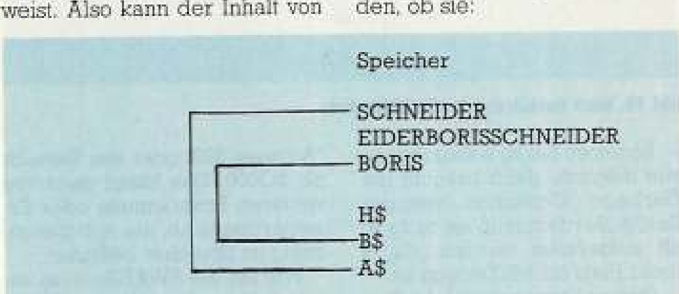


Bild 7. Der erste Schritt der Garbage Collection ist gelaufen, A$ ist aufgeräumt. In der Mitte übriggebliebener Stringmüll.

Die Garbage Collection merkt sich nun noch die Adresse, bei der der Inhalt der zuletzt bearbeiteten Stringvariablen (hier A$) endet, denn alle Stringvariablen, deren Inhalte über dieser Adresse stehen, müssen ja schon von der Garbage Collection behandelt worden sein.

Nun haben wir also A$ schon aufgeräumt, es fehlt uns noch B$. Er wird an die vorhin gemerkte Adresse (Ende von A$) geschoben, und sein Ende wird wieder gemerkt (Bild 8).

Da jetzt alle vorhandenen Stringdescriptoren über diese Grenze hinaus weisen, sind alle Stringvariablen aufgeräumt worden, und die Garbage Collection ist beendet.

Sie werden jetzt vielleicht fragen, wo denn der Haken bei der Garbage Collection liegt, denn sie war ja unwahrscheinlich schnell.

Nun, wir hatten aber auch nur zwei Strings im Speicher, die aufgeräumt werden mußten. Wenn Sie Ihren Computer mal beschäftigen wollen, so probieren Sie folgendes:

```
DIM A$(9000)
FOR I = 0 TO 9000:A$(I)="A":
NEXT I
TI$="000000":?FRE(0):?TI$
```

Bitte tippen Sie das nur, wenn die normale Speicherbelegung eingeschaltet ist, sonst erhalten Sie sofort einen OUT OF MEMORY ERROR.

Und jetzt können Sie erst mal in Ruhe diesen Artikel weiterlesen, denn Ihr Computer kann über eine Stunde lang mit der Garbage Collection beschäftigt sein! Wenn er fertig ist, sagt er Ihnen auch, wie lange er gebraucht hat.

Die Zeit, die die Garbage Collection benötigt, ist proportional zum Quadrat der vorhandenen Stringvariablen. Für die Nicht-Mathematiker unter uns: Die Garbage Collection muß so viele Durchgänge machen, wie Stringvariablen vorhanden sind, da in jedem Durchgang nur eine aufgeräumt wird. In jedem Durchgang muß aber auch jede Stringvariable angeschaut werden, ob sie:

— schon aufgeräumt ist; dann liegt ihr Descriptor über der gemerkten Grenze.

— jetzt aufgeräumt werden soll; dann hat sie den höchsten Descriptor unter der Grenze.

— erst später aufgeräumt wird, dann trifft keiner der beiden obigen Fälle zu.

Also werden soviele Vergleiche benötigt, wie die Anzahl der vorhandenen Stingvariablen im Quadrat, von der Anzahl der Speicherverschiebungen mal abgesehen, denn sie entspricht der Anzahl der Stringvariablen.

Auf einen Nenner gebracht bedeutet das, daß die FRE(0) Routine aus dem DIMA$(9000)

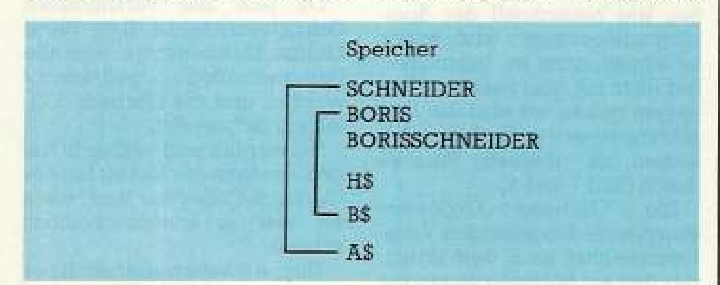


Bild 8. Die Garbage Collection ist beendet, der Restmüll am Ende des Stringspeichers kann einfach überschrieben werden.

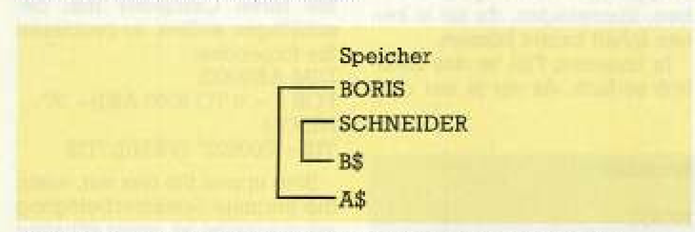


Bild 9. Und noch einmal vertauschen. Vor der Ausführung der SWAP-Routine

Beispiel 9001x9001 = 81 Millionen Vergleiche benötigt!

Diese Zahl stimmt nicht ganz mit der Wirklichkeit überein, da auch andere Faktoren eine Rolle spielen, die Größenordnungen sind aber ziemlich richtig.

Die Zeit der Garbage Collection hängt aber nicht von der Anzahl der Müllstrings ab, da diese ja gar nicht überprüft werden, sondern nur alle Descriptoren.

Es gibt mehrere Möglichkeiten, die Garbage Collection kurz zu halten:

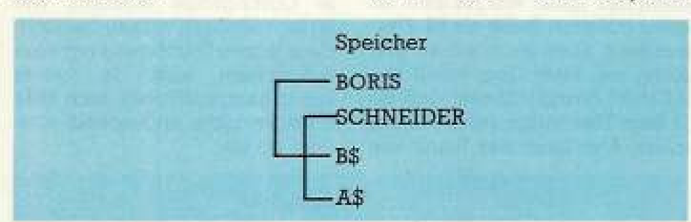


Bild 10. Nach Ausführung der SWAP-Routine.

— Benutzen Sie so wenig Strings wie möglich, dann braucht die Garbage Collection weniger Zeit, außerdem muß sie nicht so oft aufgerufen werden, da ja mehr Platz für Müllstrings ist.

— Strings wenn möglich im Programm definieren, und dann nie mehr verändern, dann entsteht weniger Stringmüll

— Weitere Stringmüll verursachende Befehlssequenzen meiden, wie zum Beispiel A$=A$ und ähnliches.

— Sollen Strings vertauscht werden, SWAP-Routine verwenden.

Halt, werden Sie jetzt sagen, was ist denn eine SWAP-Routine?

Um einen der größten Stringmüll-Verursacher, das Vertauschen von Strings, zu entschärfen, habe ich diese Routine geschrieben. Sie vertauscht zwei Strings, ohne daß Stringmüll entsteht.

Sie werden sich wahrscheinlich schon gefragt haben, warum eigentlich beim Vertauschen von Strings nicht einfach die Descriptoren der beiden Stringvariablen ausgetauscht werden. Dann könnte kein Stringmüll entstehen.

Genau das tut die SWAP-Routine. Eine ähnliche ist vielleicht schon in einer Basic-Erweiterung enthalten, die Sie benützen. Wenn nicht, so tippen Sie einfach Listing 1 ab. SWAP ist im Speicher frei verschiebbar. Sie können es also an jede beliebige Adresse legen. Sinnvoll wäre wohl der Kassettenpuffer (Adresse 828) oder der Bereich ab $C000. Das hängt ganz von weiteren Programmen oder Erweiterungen ab, die sich gleichzeitig im Speicher befinden.

Wie Sie die SWAP-Routine anwenden, steht in den REM-Zeilen des Listings, die Arbeitsweise verdeutlichen die Bilder 9 und 10.

Wenn Sie das Thema Stringverarbeitung und neue Stringfunktionen in Maschinensprache mehr interessiert: Darauf werde ich in der nächsten Ausgabe genauer eingehen, ebenso wie auf die genaue Funktionsweise der SWAP-Routine.

(Boris Schneider/gk)



Die Landkarte, der wir auf unserer Reise durch den Speicher folgen, weist diesmal — neben vielen interessanten Plätzen — auch zwei Orte auf, die nützlich sind.

Zum einen ist es die Adresse 19, zum anderen gelangen wir in den Bereich ab Speicherzelle 43, der für Speicher-Manipulationen so eminent und wichtig ist.

Adresse 18 ($12)

1. Flagge für Vorzeichen bei SIN, COS und TAN

2. Flagge bei Vergleich

Zuerst kommt das Vorzeichen der trigonometrischen Funktionen an die Reihe.

Die Routinen des Basic-Übersetzers (Interpreter), welche die drei trigonometrischen Funktionen SIN, COS und TAN berechnen, verwenden die Speicherzelle 18 zur Bestimmung des Vorzeichens.

Zur Erinnerung: Die trigonometrischen Funktionen haben in den vier »Quadranten« des Kreises (0-90, 90-180, 180-270, 270-360 Grad) nicht unbedingt dieselben Vorzeichen. Die Vorzeichen ändern sich allerdings nur an den Grenzen der Quadranten, wie in Bild 1 zu sehen ist. Die Flagge in Zelle 18 gibt das Vorzeichen nicht direkt an, sondern auf Umwegen. Die Darstellung ist in der folgenden Tabelle 1 zusammengefaßt.

WINKEL	0-90	90-180	180-270	270-360
SIN	gleich	Wechsel	Wechsel	gleich
COS	255	255	0	0
TAN	0	255	255	0

Dabei bedeutet »gleich«: 0-0-0-0 oder 255-255-255

»Wechsel«: 0-255-0-255

Da die Erklärung mit »gleich« beziehungsweise »Wechsel« nicht gerade einleuchtend ist, schlage ich vor, daß Sie sich das Ganze mit dem folgenden kleinen Programm selbst anschauen, welches für viele Werte des Winkels im Bogenmaß — und in kleinen Schritten — den Wert der Flagge, daneben den Winkel I und den Wert der Funktion mit Vorzeichen ausdruckt.

```
10 FOR I=0 TO 10 STEP 0.01
20 PRINT PEEK(18);INT(I*100)/
100;SIN(I):NEXT
```

Diese etwas umständliche Art, den Wert von I auszudrucken, vermeidet Rundungsfehler und begrenzt den Ausdruck auf zwei Dezimalstellen. Wenn Sie die Winkelwerte von I in Graden ausgedruckt haben wollen, können Sie eine andere Zeile 20 verwenden, welche die Umrechnungsformel vom Bogenmaß in Grade verwendet:

Winkel in Grad = Winkel im Bogenmaß * 180/π

```
20 Print PEEK(18);INT(I*180/π)
;SIN(I):NEXT
```

Statt SIN können Sie genauso gut COS und TAN einsetzen.

In Bild 1 sind nicht nur die Kurven und die Bereiche der Vorzeichen, sondern auch die Winkelbereiche sowohl im Bogenmaß, als auch in Graden dargestellt.

Die Speicherzelle 18 wird auch noch von anderen Routinen des Basic-Interpreters beansprucht und zwar von allen, die einen Vergleich wie < = >,)= und so weiter durchführen. Entsprechend der Art des Vergleichs steht dann in der Zelle 18 eine Ziffer von 0 bis 6.

Das folgende Programm macht das deutlich.

```
10 A=2
20 FOR I=1 TO 3
30 IF I= A THEN PRINT I;
PEEK(18);"="
40 IF I< >A THEN PRINT I;
PEEK(18);"> <"
50 IF I> A THEN PRINT I;
PEEK(18);">"
60 IF I< A THEN PRINT I;
PEEK(18);"<"
70 IF I> =A THEN PRINT I;
PEEK(18);">="
80 IF I< =A THEN PRINT I;
PEEK(18);"<="
```



mit Wandervorschlägen

Teil 3

```
90 IF I<A OR I=A THEN
PRINT I;PEEK(18);"< OR ="
100 NEXT I
```

Kurz zur Erklärung dieser Zeilen: In der FOR..NEXT-Schleife wird die Variable I mit der Konstanten A = 2 verglichen. In den Zeilen 30 bis 90 werden alle möglichen Vergleichsoperatoren durchgeprüft. Jeder der zutrifft, druckt den Wert von I, den Wert der dann in Zelle 18 stehenden Flagge und schließlich den Vergleichsoperator aus. Aus dem Resultat dieses Programms läßt sich folgende Tabelle zusammenstellen:

Vergleich	Flagge in 18
< OR =	0
> OR =	0
>	1
=	2
>=	3
<	4
<>	5
<=	6

Sie sehen, die Flagge für die kombinierten Vergleichsoperatoren entspricht der Summe ihrer Einzelwerte. Nur die Verknüpfung über OR nicht, denn die ergibt 0.

Adresse 19 ($13)
Flagge zur Kennzeichnung des laufenden Ein-/Ausgabegerätes

Immer dann, wenn von Basic Daten ein- oder ausgegeben werden, schaut die entsprechende Routine des Übersetzers in Zelle 19 nach, um welches Peripheriegerät es sich handelt. Zur Debatte stehen Tastatur, Datasette, RS232/User-Port, Bildschirm, Drucker und Floppy-Laufwerk.

Die Flagge ihrerseits ist ausschlaggebend für die feinen Unterschiede, wie zum Beispiel das Fragezeichen, bei Eingabe von der Tastatur (INPUT) oder die Anweisung »Press Play on Tape« bei Eingabe von der Datasette.

Beim Einschalten des Rechners setzt die Initialisierungsroutine des Betriebssystems, die beim VC 20 ab Adresse 58276 ($E3A4), beim C 64 ab 58303 ($E3BF) beginnt, die Flagge in Zelle 19 auf 0. Die Null bedeutet Eingabe über Tastatur und Ausgabe über Bildschirm.

Wenn Sie einen Dissassembler haben, drucken Sie doch einmal das Assemblerlisting aus. Sie werden in Adresse 58324/58325 ($E3D4/E3D5), beim C 64 in 58354/58355 ($E3F2/E3F3) den Befehl finden, der eine Null nach Zelle 19 ($13) bringt.

Immer dann, wenn ein Programm nicht Tastatur und Bildschirm, sondern eines der oben genannten anderen Peripheriegeräte anspricht (indem mit OPEN.... eine Datei = Logical File eröffnet wird), wird in Zelle 19 die Nummer der gerade bearbeiteten Datei eingetragen, mit den bereits beschriebenen Konsequenzen.

Ich will hier nicht weiter darauf eingehen, da wir den Inhalt von Zelle 19 selbst nicht auslesen können. Er wird nämlich immer gleich wieder auf Null gesetzt.

Wir können ihn aber durch POKE verändern. Durch POKE 19,1 gaukeln wir dem Rechner vor, daß Ein- und Ausgabe über »externe« Geräte läuft, selbst wenn nur die Tastatur und der Bildschirm betrieben werden.

Wenn zum Beispiel der Rechner der Meinung ist, daß ein INPUT von der Datasette kommt, druckt er kein Fragezeichen aus, auch kein EXTRA IGNORED als Fehlermeldung bei zu zahlreicher Eingabe und das alleinige Drücken der RETURN-Taste ignoriert er auch, im Gegensatz zum »normalen« INPUT. Probieren Sie es aus:

```
10 INPUT "TEST";A$
20 PRINT A$
```

In diesem Normalfall erscheint nach RUN darunter die Aufforderung TEST?

Eine Eingabe, zum Beispiel XX, erscheint mit einem Abstand daneben, und nach RETURN wird XX an den Anfang der nächsten Zeile gedruckt. Alle falschen Eingaben werden mit den üblichen Fehlermeldungen quittiert.

Jetzt fügen wir ein:

```
5 POKE 19,1
```

Nach RUN erscheint wieder die Aufforderung TEST, aber ohne Fragezeichen. Die Eingabe XX wird ohne Abstand daneben gesetzt und nach RETURN mit einem Abstand in derselben Zeile weitergeschrieben.

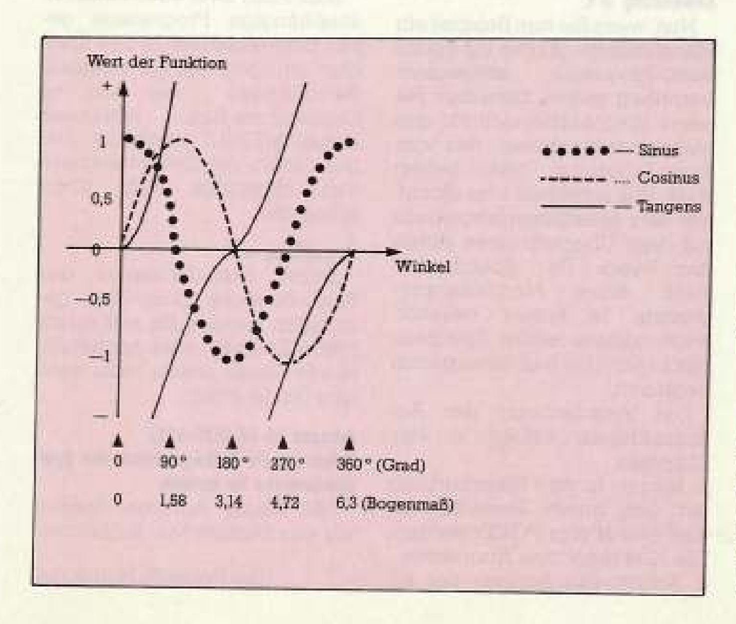
Das Drücken der RETURN-Taste setzt den Cursor nicht wie üblich in die nächste Zeile, sondern schiebt ihn in derselben Zeile weiter.

Diesen zusätzlichen Effekt muß man beachten, da er sehr störend für den Verlauf eines Programms sein kann.

Man kann ihn natürlich auch nutzbringend einsetzen, hat er doch die Eigenschaft eines automatischen »Cursor UP«. Eine pfiffige Anwendung dieser Art wurde von Brad Templeton für den PET erfunden und ist von Jim Butterfield für eine MERGE-Routine mit dem Namen »Magic Merge« veröffentlicht worden.

Da diese Routine aber primär auf der Eigenschaft der Speicherzelle 153 basiert, werde ich sie dann erläutern, sobald wir bei der Zelle 153 angelangt sind.

Zurück zur Flagge in Zelle 19. Umgekehrt können wir POKE 19,0 leider nicht nutzen, da die betroffenen Befehle GET, GET#, INPUT, INPUT# und PRINT# die Flagge sofort auf den richtigen Wert setzen. Nur PRINT und LIST tun das nicht, wie wir bei dem PRINT-Befehl oben ja gesehen haben.



Adresse 20–21 ($14–$15)
Zeilennummer für LIST, GOTO, GOSUB und ON, Zeiger der Adresse bei PEEK, POKE, SYS und WAIT

In diesen Speicherzellen wird die Zeilennummer der Sprungbefehle GOTO, ON.GOTO und GOSUB sowie die Zeilenangabe beim LIST-Befehl gespeichert. Da die Werte bis maximal 65535 gehen können, braucht der Computer 2 Bytes zur High/Low-Byte-Darstellung.

Die GOTO-Routine (im VC 20 ab 51360 = $C8A0, im C 64 ab 43168 = $A8A0) vergleicht die Zahl in 20/21 mit der laufenden Zeilenzahl. Wenn sie kleiner ist, wird ab der ersten Zeile des Programms gesucht. Ist sie aber größer, dann beginnt die Suche ab der laufenden Zeilenzahl. Die Suche geht so lange, bis die in 20/21 angegebene Zeilenzahl gefunden ist. Dann fährt das Programm mit dieser Zeile fort.

LIST speichert in 20/21 die höchste auszulistende Zeilennummer ab, falls keine Angabe beim LISTen gegeben worden ist, den Wert 65535 ($FFFF).

Die Befehle PEEK, POKE, SYS und WAIT verwenden diese Speicherzellen zur Angabe der Adressen, die dem Befehl immer folgen müssen.

Leider können wir die Speicherzellen 20/21 mit Basic-Programmen nicht bearbeiten; ihr Inhalt wird immer gleich auf 20 zurückgesetzt.

Adresse 22 ($16)
Zeiger auf den nächsten freien Speicherplatz im »Temporary String Descriptor Stack«

Dieser Zeiger bezieht sich in seiner Wirkung auf die übernächsten Speicherzellen 25 bis 33 ($19-$21).

Diese werden als Stapelspeicher (Stack) für Angaben über vorläufige Zeichenketten — auf englisch »Temporary String Descriptor« — verwendet (siehe unten).

Die Speicherzelle 22 ($16) ihrerseits enthält einen Zeiger auf den jeweils nächsten verfügbaren Platz in diesem Speicher ab Zelle 25. Da er eine Kapazität von 3 mal 3 Byte hat, zeigt der Zeiger auf die Zelle 25 ($19), wenn er leer ist. Bei einem Eintrag zeigt er auf 28 ($1C), bei zwei Einträgen auf 31 ($1F) und schließlich auf 34 ($22), wenn der Speicher voll ist.

Eine Zeichenkette ist dann »vorläufig«, wenn sie noch nicht einer Stringvariablen zugeordnet worden ist, zum Beispiel

MEMORY MAP mit Wandervorschlägen Teil 3



»Mahlzeit« in dem Basic-Befehl PRINT "MAHLZEIT.

Beim Einschalten setzt das Betriebssystem mit der Einschalt-Routine ab Adresse 58303 ($E3BF) im C 64, beim VC 20 ab 58276 ($E3A4) den Zeiger auf 25. Die Stringverwaltungsroutine ab 46215 ($B487) im C 64 beziehungsweise ab 54407 ($D487) im VC 20 fragt bei String-Eingaben die Flagge ab. Nach jeder Eintragung in den Speicher ab Zelle 25 wird der Zeiger um 3 weitergesetzt.

Sie können die Leerflagge 25 mit PRINT PEEK (22) leicht nachprüfen.

Die anderen Eintragungen können nicht nachgeprüft werden, weil sie sofort auf 25 zurückgesetzt werden.

Wir können sie aber durch POKE beeinflussen; ob das sinnvoll ist, ist eine andere Frage.

```
10 POKE 22,34
20 PRINT "MAHLZEIT"
```

Die Zahl 34 in Zelle 22 sagt dem Programm, daß der Speicher ab Zelle 25 voll ist und wir bekommen statt der MAHLZEIT eine Fehlermeldung serviert.

Mit einem POKE-Befehl, der als Argument die für den vorgesehenen Zweck ungültige Zahl 35 verwendet:

POKE 22,35

erreichen wir allerdings zwei interessante »Dreckeffekte«. Zum einen unterdrückt der Befehl die Ausgabe des READY, zum anderen aber bewirkt er, daß bei LIST ein Listing ohne Zeilennummern ausgedruckt wird, sowohl auf dem Bildschirm als auch mit dem Drucker.

Die Idee dazu habe ich von Mike Apsey's Hinweis in »Commodore User« Juli 1984. Mit Zeilen-
ummern versehen, läßt sich je-
beliebige Text schreiben,
ssern, verschieben, ab-
ern, aber nicht RUNen!
OKE-Befehl von oben (POKE 22,35) gefolgt von einem CMD und LIST, druckt dann alles brav als reinen Text aus. Die maximale Zeilenlänge entspricht der Zeilenlänge des jeweiligen Computers.

Probieren Sie es aus:

```
10 DER COMPUTER BIETET
IN DER
20 DATENFERNÜBERTRA-
GUNG
30 UNGEAHNTE MÖGLICH-
KEITEN.
40 ABER DIE GEFAHR
50 USW. USW.
60:
```

Jede Zeile wird mit der RETURN-Taste abgeschlossen. Damit auch alles gedruckt wird, muß — zumindest bei meinem Drucker (1526) — eine »Leerzeile« folgen (Zeile 60). Mit

POKE 22,35:OPEN 1,4:CMD 1:LIST

wird der Text ohne Zeilennummern ausgedruckt. Sie können ihn vorher nach Belieben verändern.

Wie gesagt, nur nicht mit RUN starten, denn das bringt unweigerlich eine Fehlermeldung.

Adresse 23-24 ($17-$18)
Zeiger auf die Adresse der letzten Zeichenkette im »Temporary String Stack«

Der Inhalt dieser zwei Bytes zeigt auf den zuletzt benützten Speicherplatz innerhalb der Adresse 22 bis 33. Das heißt, daß der Wert in 23 ($17) immer um 3 kleiner ist als der in 22 ($16), während der Wert in 24 ($18) eine Null ist.

Adresse 25-33 ($19-$21)
Stapelspeicher für Angaben über vorläufige Zeichenketten

Das ist also der Speicherbereich, von dem in den beiden vorigen Abschnitten dauernd die Rede war. Ich gebe zu, »Descriptor Stack for Temporary Strings« drückt die Sache präziser aus als der deutsche Text.

Die Bedeutung eines »vorläufigen« Strings habe ich oben in der Beschreibung der Speicherzelle 22 erklärt.

Was ein Stapelspeicher (Stack) ist, entnehmen Sie bitte dem Texteinschub. Jeder der drei Byte langen Angaben im Stack von 22 bis 33 enthält die Länge sowie die Anfangs- und Endadressen eines vorläufigen Strings, ausgedruckt als Verschiebung im Basic-Speicherbereich.

Adresse 34-37 ($22-$25)
Verschiedene Zwischenspeicher

Diese vier Speicherzellen werden vom Basic-Übersetzer (Interpreter) für verschiedene Zwischenergebnisse und Flaggen benützt, die aber dem Programmierer nichts nützen.

Adresse 38-42 ($26-$2A)
Arbeitsspeicher für arithmetische Operationen

Diese Speicherzellen werden von den Basic-Routinen bei der Multiplikation und Division als »Notizblatt« verwendet. Auch die Routinen, welche die erforderliche Speichergröße beim Definieren eines Zahlenfeldes (Array) ausrechnen, benutzen diesen Bereich.

Adresse 43-44 ($2B-$2C)
Zeiger auf den Anfang der Basic-Programme im Speicher

Dieser Zeiger, in der Low/High-Byte-Darstellung, gibt dem Basic-Übersetzer an, ab welcher Speicherzelle das Basic-Programm beginnt. Normalerweise ist diese Adresse fest vorgegeben. Beim C 64 zum Beispiel zeigt der Zeiger auf 2049 ($801). Beim VC 20 ist die Lage schon schwieriger, denn der Speicherbeginn hängt davon ab, welche Speichererweiterung eingesetzt ist. Die folgende Tabelle gibt darüber Auskunft.

Beginn des Programmspeichers	
C 64	2049 ($801)
VC 20 (GV)	4097 ($1001)
VC 20 (+3 K)	1025 ($401)
VC 20 (+ 8 K)	4609 ($1201)

Mit dem Befehl PRINT PEEK (43) + PEEK (44)*256 läßt sich der jeweilige Beginn des Programmspeichers leicht feststellen. Mit einem POKE-Befehl kann der Programmierer diese Anfangsadresse verändern. Wozu das gut ist, fragen Sie?

Anwendung #1:

Nun, wenn Sie zum Beispiel ein Maschinenprogramm mit einem Basic-Programm gemeinsam betreiben wollen, brauchen Sie einen Speicherbereich für das Maschinenprogramm, der vom Basic-Programm nicht belegt wird. Wir sprechen vom »Schützen des Maschinenprogramms vor dem Überschreiben durch das Basic«. Der Speicherbereich eines Maschinenprogramms ist immer bekannt. Nach seinem letzten Speicherplatz kann das Basic-Programm beginnen.

Die Verschiebung der Anfangsadresse erfolgt in vier Schritten:

1. Schritt: In den Speicherplatz vor dem neuen Basic-Bereich muß eine Null gePOKEt werden. Die Null dient zum Abgrenzen.
2. Schritt: Die Adresse der ersten Speicherzelle wird in die Low-High-Byte-Darstellung umgerechnet. Ich verweise dazu auf die Erklärung dieses Vorgangs in der ersten Folge dieses Kurses in Ausgabe 11/84.
3. Schritt: Das Low-Byte wird in die Speicherzelle 43, das High-Byte in die Zelle 44 gePOKEt.
4. Schritt: Die Operation muß unbedingt mit dem Befehl NEW abgeschlossen werden, um sicherzustellen, daß der neue Basic-Bereich auch »sauber« ist.

Im folgenden kleinen Programm wird angenommen, daß der Speicher bis zur Adresse 5000 ($1388) durch ein Maschinenprogramm belegt ist. Das Basic-Programm kann daher ab 5002 ($138A) anfangen, denn in 5001 muß ja eine Null stehen. Die Adresse 5002 teilt sich auf in ein High-Byte von INT (5002/256) = 19 und ein Low-Byte von 5002—(19*256) = 138.

```
10 POKE 5001,0
20 POKE 43,138
30 POKE 44,19
40 NEW
```

Der Effekt einer solchen »Verbiegung« des Zeigers in 43/44 wird im Texteinschub 2 »Der sichtbare Basic-Speicher« demonstriert.

Neben der oben erwähnten Anwendung der Zeigerverbiegung gibt es noch andere Möglichkeiten:

Anwendung #2:

Christoph Sauer hat in seinem Kurs »Der gläserne VC 20« in Ausgabe 10/84 auf Seite 158 gezeigt, wie man mehrere Programme gleichzeitig im Speicher unterbringen und zwischen ihnen umschalten kann.

Anwendung #3:

Man kann zwei oder mehrere unabhängige Programme genau hintereinander in den Speicher bringen, um sie aneinanderzuhängen , was dem im Commodore-Basic fehlenden Befehl MERGE entspricht. Dabei dürfen die Zeilennummern sich allerdings nicht überschneiden.

Anwendung #4:

Durch Hinaufschieben des Basic-Bereichs kann Platz geschaffen werden für selbstdefinierte Zeichen oder hochauflösende Grafik (siehe dazu auch den Grafik-Kurs).

Adresse 45-46 ($2D-$2E)
Zeiger auf die Anfangsadresse des Speicherbereichs für Variable

Mit dieser Adresse werden wir das nächste Mal fortfahren.

(Dr. Helmuth Hauck/aa)

Was ist ein Stapelspeicher (Stack)?

Texteinschub #1

Der normale Arbeitsspeicher des Computers, auf englisch »Random Access Memory« oder kurz RAM genannt, hat für jede Speicherzelle eine eigene Adresse, die beim Schreiben in den Speicher oder beim Lesen aus dem Speicher angegeben werden muß.

Als Analogie möge eine Aktenablage dienen, bei der jeder Akt (Brief, Papier, Zeichnung) in einen Ordner kommt, mit Nummer versehen.

Um einen Akt herauszuholen, muß man die Nummer (Adresse) kennen, unter der er abgelegt ist.

Ein Stapelspeicher, auf englisch »Stack« genannt, funktioniert wie eine Aktenablage, bei der jeder Akt einfach oben auf einen Stapel gelegt wird, daher der Name. Diese Ablage erfolgt ohne Kennzeichnung oder Nummer, einfach immer der Reihe nach.

Einen Akt kann man aus einem Stapelspeicher nicht beliebig herausholen, da immer nur der oberste Akt zugänglich ist.

Die Methode der Stapelspeicher bietet sich überall dort an, wo es auf die Reihenfolge der gespeicherten Daten ankommt. Basic merkt sich zum Beispiel der Reihe nach die Adressen, von denen aus mit GOSUB ein Unterprogramm angesprungen wird. Wenn mehrere GOSUBs hintereinander eingesetzt werden, liegt auf dem Stapel immer die letzte Absprungadresse bereit zum Rücksprung.

Ein Stapelspeicher hat demnach nur eine einzige Adresse, die sowohl zum Abspeichern als auch zum Auslesen dieselbe ist.

Voraussetzung eines Stapelspeichers ist natürlich eine Routine, welche alle gespeicherten Daten im Stapelspeicher um einen Platz weiterschiebt, wenn eine neue Information »oben auf den Stapel gelegt wird«.

Das Basic der Commodore-Rechner verwendet mehrere dieser Stapelspeicher.

Die Programmiersprache Forth ist völlig auf dem Prinzip des Stapelspeichers aufgebaut.

Der sichtbare Basic-Speicher

Wenn wir den Variablen A die Adresse des Speicherbeginns der Basic-Programme zuordnen und dann mit einer FOR...NEXT-Schleife den Inhalt dieser und der nächsten 100 Speicherplätze ausdrucken, sehen wir in dezimaler Darstellung die ersten 101 Zahlenwerte, mit denen der Computer ein Basic-Programm abspeichert.

Ein Verbiegen des Zeigers in Speicherzelle 43/44 kann auf diese Weise in seiner Wirkung sichtbar gemacht werden.

Als Demo-Programm wähle ich zwei Zeilen, welche die Zahlen 1 bis 9 und die Buchstaben A bis I ausdrucken.

```
10 PRINT "123456789"
20 PRINT "ABCDEFGHI"
100 A=2049 : REM*C 64
      4097 : REM*VC 20 ohne Erweiterung
      1025 : REM*VC 20 mit 3 KByte
      4609 : REM*VC 20 mit 8 KByte oder mehr
110 PRINT CHR$(147)
```

Zeile 100 definiert den Speicheranfang. Zeile 110 löscht den Bildschirm.

```
120 FOR J=A TO A+100
130 PRINT PEEK (J);
140 NEXT J
```

Die Befehle in den Zeilen 120 bis 140 drucken den Inhalt der ersten 101 Zellen dieses Basic-Programms aus. SAVEn Sie bitte dieses kleine Programm, denn wir brauchen es noch einmal. Dann geht es los mit RUN. In Bild 2 ist der Bildschirm-Ausdruck des VC 20 mit 8 KByte dargestellt, der des C 64 zeigt praktisch dieselbe Information.

READY.

Bild 2. Ausdruck vom VC 20 mit 8 KByte

Überspringen Sie bitte zunächst die ersten beiden Zahlen. Die dritte und vierte Zahl ist 10 und 0. Das ist (als Low- und High-Byte) die Nummer der ersten Zeile des Basic-Programms. Dann folgt 153, das ist der interne Codewert für PRINT. Diese Codes für alle Basic-Befehlswörter heißen »TOKEN« und sind zusammen mit den ASCII-Codes aller Zahlen, Zeichen und Funktionen in den Commodore-Handbüchern angegeben.

Die nächste Zahl im Bildschirm ist die 34, sie ist der ASCII-Code für den Gänsefuß. Danach folgen in aufsteigender Reihenfolge die ASCII-Codes der Ziffern 1 (48) bis 9 (57). Danach sehen Sie wieder den Gänsefuß (34). Schließlich kommt eine Null als Abstandszeichen zur nächsten Basic-Zeile.

Machen Sie bitte folgendes Experiment: Ausgehend von der Adresse der ersten auf dem Bildschirm ausgedruckten Speicherzellen — zum Beispiel 4609 beim VC 20 mit 8 KByte — zählen Sie die Zellen weiter bis zur Abgrenzungs-Null. In meinem Beispiel steht die 0 in Zeile 4625. Das heißt, daß die nächste Basic-Zeile in 4626 anfängt. Und das ist genau die Zahl, die in den ersten beiden Zellen steht, die wir vorhin übersprungen haben; in meinem Beispiel steht da 18 18. Machen wir die Probe: 18 + 256 x 18 = 4626.

Jede Basic-Zeile im Speicher beginnt also mit der Adresse der nächsten Zeile (sie heißt Koppeladresse) und endet mit einer Null. Ab 4626 folgt dann die nächste Koppeladresse, danach mit 20 0 die Zeilennummer, und Sie erkennen jetzt sicher die Codes der Angaben von Zeile 20 wieder.

So, jetzt wollen wir den Zeiger in 43/44 verbiegen. Ich schlage vor, daß wir den Basic-Beginn um zehn Adressen höher schieben wollen. Sie müssen jetzt die in Zeile 100 oben verwendete Zahl für A in die High/Low-Byte-Darstellung umrechnen und das Low-Byte um 10 erhöhen. Dieses Zahlenpaar POKEn wir in die Zellen 43/44. Vorher müssen wir aber noch in die Zelle (A+10)—1 eine Abstands-Null POKEn.

Wir geben diese Befehlssequenz im Direktmodus ein:

☐ für den C 64:
POKE 2058,0:POKE 43,11:POKE 44,8:NEW
☐ für den VC 20 (GV):
POKE 5006,0:POKE 43,143:POKE 44,19:NEW
☐ für den VC 20 (= 3 KByte):
POKE 1034,0:POKE 43,11:POKE 44,4:NEW
☐ für den VC 20 (> 8 KByte)
POKE 4618,0:POKE 43,11:POKE 44,18: NEW

Jetzt ist der Anfang des Basic-Speichers versetzt. Um das zu prüfen, geben wir das kleine Programm von oben nochmal ein und lassen es mit RUN laufen. Der resultierende Bidlschirmausdruck ist in Bild 3 dargestellt.

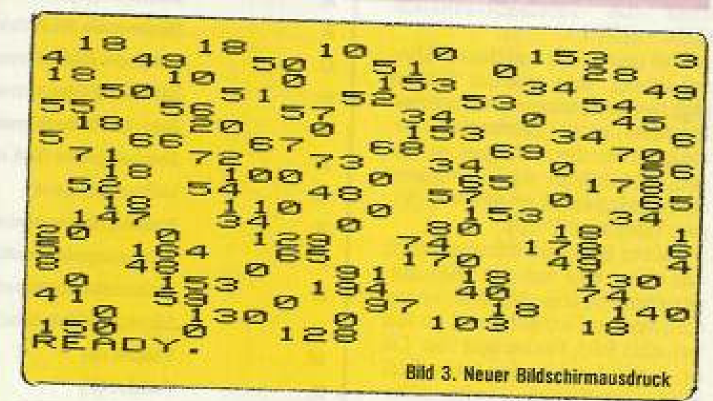


Bild 3. Neuer Bildschirmausdruck

Die ersten Zahlen sind genauso wie vorher. Es sind auch die Reste von vorher, da wir den Speicher nicht auf Null gesetzt haben. Aber zählen Sie bitte die ersten zehn Adressen hoch. Da finden Sie unser Programm von vorhin genau wieder, beginnend mit der Abstandsnull. Aber Vorsicht, lassen Sie sich nicht verwirren, denn die Koppeladressen sind natürlich jetzt auch jeweils um 10 höher. Aber hinter den Koppeladressen finden wir wieder unser Programm, in gleicher Weise dargestellt wie beim ersten Mal. Da der Zeiger in 43/44 von allen entsprechenden Routinen des Übersetzers und des Betriebssystems abgefragt wird, läuft ein verschobenes Programm fehlerfrei, solange natürlich der Zeiger nicht wieder verändert wird.

Der gläserne VC 20

Teil 4

Nachdem wir in der letzten Folge Funktion und Sinn verschiedener Vektoren erläutert haben, wenden wir uns heute einem nicht minder interessanten Kapitel, nämlich den verschiedenen hochauflösenden Grafikmodi zu.

Dieser Teil unserer Serie soll schrittweise zur Gestaltung des Bildschirmes in hochauflösender beziehungsweise vielfarbiger Grafik hinführen. Dazu ist eine genaue Kenntnis der VIC-Register und des Speicheraufbaus notwendig. Wer sich nochmals eine Übersicht über die Aufteilung des Speichers verschaffen will, der kann jetzt in Teil 1 und 3 dieser Serie nachschlagen.

Der VC 20-Zeichensatz

Wenn man den Computer einschaltet, so stehen »serienmäßig« zwei Zeichensätze zur Verfügung (Großschreibung mit Grafikzeichen oder Groß- und Kleinschreibung). Die Umschaltung zwischen beiden erfolgt, wie bekannt, unter anderem über die Tastenkombination CBM+SHIFT.

Was geschieht bei dieser Umschaltung, und warum lassen sich beide Zeichensätze (normalerweise) nicht gleichzeitig darstellen? Um diese Fragen zu klären, betrachten wir im folgenden das Innenleben des VIC (Video Interface Chip).

Dieser Baustein kümmert sich um alles, was mit der Informationsweitergabe an das angeschlossene Fernsehgerät zu tun hat, also Bild, Farbe und Ton. Da er — wie bereits das letzte Mal kurz angedeutet — eigenständig, das heißt unabhängig von der CPU arbeitet, muß der VIC selbständig auf Informationen zurückgreifen können. Diese sind zum einen in den internen Registern, zum anderen in den normalen Speicherstellen enthalten. Die Register speichern Parameter für die Bildschirmgröße, die Tongeneratoren, die Lage des Bildschirm-, Farb- und Zeichenspeichers. Ferner können aus ihnen die Zustandswerte des Paddels und des Lichtgriffels ausgelesen werden (Tabelle 1).

Register	Registerfunktion	Normalwert
0. 36864	ABBB BBBB	5
1. 36865	CCCC CCCC	25
2. 36866	HDDD DDDD	150
3. 36867	GEEE EEEF	46 oder 176
4. 36868	GGGG GGGG	
5. 36869	HHHH IIII	240
6. 36870	KKKK KKKK	0
7. 36871	LLLL LLLL	0
8. 36872	MMMM MMMM	255
9. 36873	NNNN NNNN	255
10. 36874	JRRR RRRR	0
11. 36875	OSSS SSSS	0
12. 36876	PTTT TTTT	0
13. 36877	QUUU UUUU	0
14. 36878	WWWW VVVV	0
15. 36879	XXXX YZZZ	27

A:	Einblendungsmodus
B:	Horizontale Bildschirmzentrierung
C:	Vertikale Bildschirmzentrierung
D:	Zahl der Bildschirmspalten
E:	Zahl der Bildschirmzeilen
F:	Zeichengröße (8x8 oder 8x16)
G:	Lichtgriffelraster
H:	Bildschirmspeicheradresse
I:	Zeichenspeicheradresse
K:	Lichtgriffel (horizontal)
L:	Lichtgriffel (vertikal)
M:	Paddel (X)
N:	Paddel (Y)
J,O,P,Q:	Tongenerator 1-4 ein/aus
R,S,T,U:	Frequenz Tongenerator 1-4
V:	Lautstärke
W:	Hilfsfarbe
X:	Bildschirmfarbe
Y:	RVS-Modus
Z:	Rahmenfarbe

Tabelle 1. Die einzelnen Registerbelegungen im VIC

Nun wollen wir sehen, wie man die Startadressen der einzelnen VIC-externen Speicherstellen ermitteln kann. Beginnen wir mit dem Videospeicher.

Wenn man Bild 1 betrachtet merkt man schon, wie haarig die Berechnung mit Hilfe der einzelnen Bits ist. Aus den beiden hier beteiligten Registern Nummer (#) 2, Adresse 36865, und Nummer 5 (Adresse 36869) müssen jeweils bestimmte Bits für die Erstellung der Adresse herangezogen werden. Bild 1 zeigt nun, wie die jeweiligen Bits (die vier höchstwertigen aus Register #5 und Bit 7 aus Register #2) in das High-Byte-Schema »eingebaut« werden. Ferner muß Bit 7 aus Register #2 invertiert werden, das heißt aus der Null wird eine Eins und umgekehrt.

Fassen wir noch einmal zusammen: Aus bestimmten Bits der Register 2 und 5 entsteht das High-Byte der Bildschirmspeicheranfangsadresse (ein Low-Byte gibt's hier nicht). Außerdem sind bestimmte Bits aus diesem Adreßschema immer auf Null. Die Konsequenz daraus ist, daß der Videobereich nur in bestimmten Speicherbereichen angesiedelt werden kann. Tabelle 2 zeigt die möglichen Adressen, die sich aus Bild 1 ergeben. Zu beachten ist, daß das untere Nibble (1 Nibble = 4 Bits) im Register #5 dem Zeichenspeicher zugeordnet ist, dieses darf vorläufig noch nicht angetastet werden.

Kompliziertes — Die Adreßermittlung

In gleicher Weise wie für den Bildschirm wird auch die Adresse des Farbspeichers ermittelt.

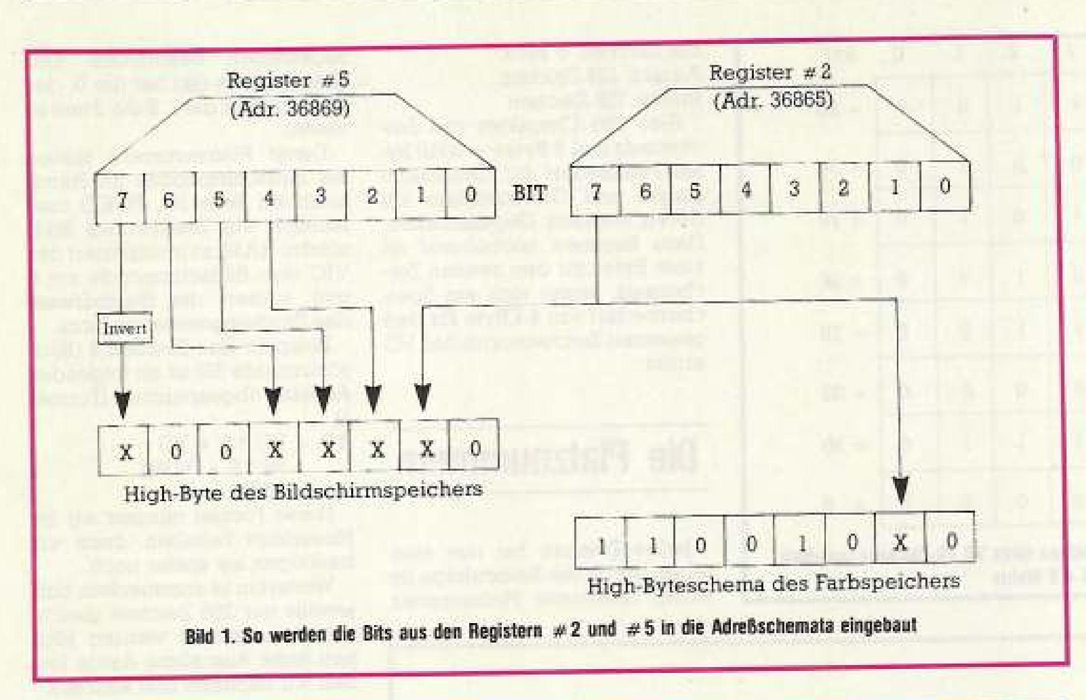


Bild 1. So werden die Bits aus den Registern #2 und #5 in die Adreßschemata eingebaut

Register #5 Bit 4-7	Register #2 Bit 7	Adresse	Bemerkung
1000	0	0	Zeropage (nicht nutzbar)
1000	1	512	Stack (nicht nutzbar)
1001	0	1024	
1001	1	1536	
1010	0	2048	
1010	1	2560	3 KByte Erweiterung
1011	0	3073	
1011	1	3584	
1100	0	4096	
1100	1	4608	
1101	0	5120	
1101	1	5632	Grundversions-speicher
1110	0	6144	
1110	1	6656	
1111	0	7168	
1111	1	7680	

Tabelle 2. Diese Tabelle zeigt die möglichen Bildschirmspeicherstellen

Register #5
(Adr. 36869)
7 6 5 4 3 2 1 0 BIT
Invert
X 0 0 X X X 0 0
High-Byteschema des Zeichenspeichers

Bild 2. Auch die Anfangsadresse des Zeichengenerators wird mit Hilfe eines Adreßschemas ermittelt

Auch hier gibt es ein Adreßschema (Bild 1), in das Bit 7 aus Register #2 eingesetzt wird. Daher erklärt sich auch, warum sich bei einer Erweiterung von mehr als 8 KByte der Farbspeicher verschiebt. Liegt der Bildschirmspeicher bei Adresse 4096, dann hat Bit 7 aus Register #2 den Wert Null, wodurch sich eine Coloradresse von 37888 ergibt. Im anderen Fall (Bildschirmstartadresse 7680) ist genau dieses Bit auf eins, wodurch es eine Verschiebung nach 38400 gibt.

Kommen wir nun zum Zeichenspeicher. Er erhält Informationen über jedes Zeichen, das auf dem Bildschirm darstellbar ist. Auch hier kann über das untere Nibble von Register #5 die Startadresse ermitteln. (Bild 2). Wenn die Bits 0-3 auf Null gesetzt sind, liegt sie bei 32768, also im Zeichengenerator-ROM (Normalstellung). Dort sind alle beim VC 20 verfügbaren Zeichen abgelegt (in welcher Form dies geschieht, werden wir später noch sehen).

Wie sie bestimmt schon bemerkt haben, spielt bei der Verwaltung des Bildschirms die Adresse 36869 (Register #5) eine bedeutende Rolle. Fragt man sie mit PEEK ab, so erhalten wir in der Grundversion oder bei einer 3-KByte-Erweiterung den Wert 240, bei einem um 8 KByte erweiterten Speicher 192. Zerlegt man diese Zahlen wie in Bild 2, so erhält man jedes Mal die gleiche Adresse für den Zeichenspeicher. Setzen wir nun Bit 1 auf 1 (= 242 in GV beziehungsweise 194 bei 8 KByte), liegt die Anfangsadresse des Zeichenspeichers bei Adresse 34816 (Tabelle 3), wodurch wir auf einmal Groß- und Kleinschrift auf dem Bildschirm haben. Die eine der zwei zu Anfang gestellten Fragen haben wir damit beantwortet.

Wie sind die im Zeichengenerator abrufbaren Zeichen eigentlich aufgebaut? Spätestens hier müssen wir ein Zeichen (im wahrsten Sinne des Wortes) unter die Lupe nehmen (Bild 3). Die vergrößerte Darstellung läßt erkennen, daß jedes Zeichen aus acht Zeilen mit je acht Spalten (8 x 8 Matrix) zusammengesetzt ist. Innerhalb dieses Gitters sind einzelne Punkte gesetzt beziehungsweise gelöscht. Jede Zeile innerhalb dieser Matrix ist als Byte (mit verschieden gesetzten Bits) im Zeichengenerator-ROM abgelegt (ROM natürlich deshalb, damit die Zeichen nach dem Abschalten erhalten bleiben).

Punkt für Punkt — der Zeichenaufbau

Im unteren Teil von Bild 3 sieht man sehr gut, daß ein gesetzter Punkt der 1, ein nicht gesetzter (gelöschter) Punkt der 0 entspricht. Die Zeile wird dann in einen normalen Zahlenwert zwischen 0 und 255 umgerechnet. In dieser Form und in der Zeilenreihenfolge von oben nach unten sind sie nun im ROM abgelegt. Der »Klammeraffe« @ (Bildschirmcode 0) ist als erstes Zeichen dort zu finden. Folglich enthält die Startadresse des Zeichengenerators (Adresse 32768) die 0-te Zeile des Zeichens — also 28, Adresse 32769 enthält 34 und so weiter. Damit machen wir folgende Rechnung auf:

Zeile	7.	6.	5.	4.	3.	2.	1.	0. BIT	
0.	0	0	0	1	1	1	0	0	= 28
1.	0	0	1	0	0	0	1	0	= 34
2.	0	1	0	0	1	0	1	0	= 70
3.	0	1	0	1	0	1	1	0	= 86
4.	0	1	0	0	1	1	0	0	= 76
5.	0	0	1	0	0	0	0	0	= 32
6.	0	0	0	1	1	1	1	0	= 30
7.	0	0	0	0	0	0	0	0	= 0

Bild 3. Diese Grafik zeigt den Aufbau eines VC 20-Zeichens innerhalb der 8 x 8 Matrix

Ein Zeichen: 8 Byte
Anzahl: 128 Zeichen
Invers: 128 Zeichen

Also 256 Charakter pro Zeichensatz mal 8 Bytes = 2048 Bytes Platzbedarf für Großbuchstaben und Grafikzeichen mit deren inversen Gegenstücken. Dazu kommen nocheinmal so viele Bytes für den zweiten Zeichensatz, womit sich ein Speicherbedarf von 4 KByte für den gesamten Zeichenvorrat des VC ergibt.

Die Platznummern

Jedes Zeichen hat nun eine (eben durch die Reihenfolge im ROM) bestimmte Platznummer zugewiesen bekommen. Der Klammeraffe (@) hat die 0, der Buchstabe A die 1, B die 2 und so weiter.

Diese Platznummern stehen als Bildschirmcodes im Handbuch auf Seite 141. POKEt man nämlich ein Zeichen ins Bildschirm-RAM, so multipliziert der VIC den Bildschirmcode mit 8 und addiert die Startadresse des Zeichengenerators hinzu.

Beispiel: Das Zeichen $ (Bildschirmcode 36) ist ab folgender Adresse abgespeichert (Formel 1).

AD = BC * 8 + ZG
36 * 8 + 32768
33056

Diese Formel müssen wir im Hinterkopf behalten, denn wir benötigen sie später noch.

Weiterhin ist anzumerken, daß jeweils nur 256 Zeichen gleichzeitig dargestellt werden können (eine Ausnahme davon lernen wir nächstes Mal kennen).

Register #5 Bits 0-3	Adresse	POKE 36869, (GV)	(8KByte)	Belegung
0000	32768	240	192	Zeichen ROM
0001	33792	241	193	Inverse Großschrift
0010	34816	242	194	Groß-, Kleinschrift
0011	35840	243	195	Inverse Kleinschrift
0100	36864	244	196	
0101	37888	245	197	Nicht verwendbar, da belegt
0110	38912	246	198	
0111	39936	247	199	
1000	0	248	200	Zeropage
1001	1024	249	201	
1010	2048	250	202	3 KByte Erweiterung
1011	3072	251	203	
1100	4096	252	204	
1101	5120	253	205	Grundversion
1110	6144	254	206	
1111	7168	255	207	

Tabelle 3. Die möglichen Zeichengeneratoranfangsadressen (jedoch sind nicht alle möglichen Adressen auch sinnvoll)

Characterstyling: Eigene Sonderzeichen

Schön wäre es nun, wenn man jedes Zeichen nach eigenen Wünschen abändern könnte. Dies geht (wie sich inzwischen wohl herumgesprochen hat), indem man den Zeichengenerator (also die Zeicheninformationen) ins RAM verlegt und das Register #5 ändert.

Dazu müssen wir uns anhand einer Grafik des Speicheraufbaus (Bild 4) überlegen, wo wir diese Zeichen abspeichern, denn wie in Tabelle 2 zu sehen ist, lassen sie sich nicht überall, sondern nur an bestimmten Stellen unterbringen. Auch hier muß man zwischen den beiden Aus-

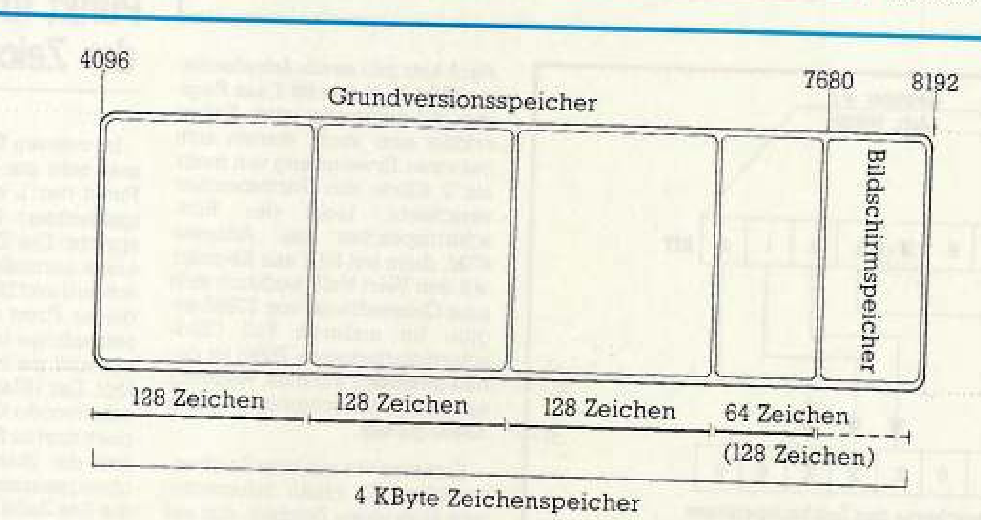


Bild 4. So können die 512 Zeichen im Grundversions-RAM verteilt werden

baustufen Grundversion /+3 KByte und ab 8 KByte unterscheiden.

Nachhilfe — Die logischen Funktionen

Aufgrund der Zweiteilung des Registers #5 sollte man alle Änderungen nur mit Hilfe logischer Funktionen durchführen, dazu ein kleiner Exkurs.

Gerade bei unserem Fall, aber auch bei anderen Gelegenheiten, will man nur bestimmte Bits einer Speicherstelle ändern. Mit den boolschen Operationen OR und AND lassen sich jeweils die gewünschten Bits Ein- beziehungsweise Ausschalten.

Die erste wichtige Verknüpfungsoperation ist die ODER- (in Basic OR) Operation, die zwei Bits nach der in Tabelle 4a abgebildeten Wahrheitstabelle verknüpft. Haben wir nun eine Kette von 8 Bit (also ein Byte), so kann man mit Hilfe dieser Operation bestimmte Bits »einschalten«. Dazu ein Beispiel: Von einer Speicherstelle sollen die Bits 1 und 6 eingeschaltet werden. Da Basic die Binärzahlen nicht direkt, sondern nur als Dezimalzahlen verarbeiten kann, müssen wir zunächst alle Werte in dieses Zahlensystem umwandeln.

Bit	7	6	5	4	3	2	1	0
Wertigkeit	128	64	32	16	8	4	2	1
	0	1	0	0	0	0	1	0

Durch Addition der Wertigkeiten der zu setzenden Bits ergibt sich der Wert 66 (= 64+2). Durch die OR-Verknüpfung der Speicherstelle mit 66 ergibt sich folgende Bitstruktur (für den ursprünglichen Inhalt der Speicherstelle nehmen wir einmal 229 an):

Speicherstelle	1	1	1	0	0	1	0	1	=229
Bit 1+6 setzen:	0	1	0	0	0	0	1	0	=66
229 OR 66:	1	1	1	0	0	1	1	1	=231

Der gewünschte Effekt ist eingetreten, das heißt die Bits 1 und 6 wurden gesetzt (da Bit 6 bereits eingeschaltet war, ergab sich hier keine Änderung).

Analog verfährt man beim Löschen bestimmter Bits. Dazu wird dann allerdings die UND-Operation (Basic-Befehl AND) verwendet. Wie Tabelle 4b zeigt, bleiben nur die Bits unverändert, die mit 1 »verANDet« werden. Auch hierzu ein Rechenbeispiel, bei dem die Bits 1 und 6 gelöscht werden sollen:

Speicherstelle:	1	1	1	0	0	1	1	1	=231
Bit 1+6 löschen:	1	0	1	1	1	1	0	1	=189
231 AND 189:	1	0	1	0	0	1	0	1	=165

Auch hierbei muß man sich vor der Verknüpfung ein Bitmuster berechnen, in dem die zu löschenden Bits auf Null, die nicht zu verändernden auf 1 gesetzt werden müssen. Wem dies noch nicht restlos klar ist, der sollte anhand von einigen Rechenbeispielen diese Operationen üben, denn wir benötigen diese in der nächsten Folge, wenn es darum geht, jeden Grafikpunkt einzeln anzusteuern (beispielsweise über Koordinaten).

Aber auch das Register #5 kann damit wesentlich eleganter geändert werden. So schaltet POKE 36869,PEEK (36869) OR 1 auf Groß-/Kleinschreibung um (überlegen Sie mal, warum dies so ist), wobei man keine Unterscheidung zwischen verschiedenen Speicherausbaustufen treffen muß.

Fassen wir das bis jetzt behandelte noch einmal kurz zusammen: Wir haben also die Möglichkeit, den gesamten Zeichensatz ins RAM zu verlegen. Dort kann er dann nach eigenen Wünschen verändert werden. Bevor man jedoch irgendeine Änderung vornehmen kann, müssen die Zeichen in das bisher ja »leere« RAM kopiert werden.

Das in Listing 1 abgedruckte Basic-Programm erfüllt diese Aufgabe. Die Routine ist so aufgebaut, daß lediglich die Variable RG eingesetzt werden muß. Sie wurde mit Absicht so allgemein gehalten, damit sie für alle Speichererweiterungen einsetzbar ist.

Hier nun eine Auflistung der einsetzbaren Werte (sie entsprechen im übrigen denen in Tabelle 2):

RG = 12: Aus diesem Registerwert errechnete sich das Programm die Zeichenspeicheradresse 4096. Die Zeichen füllen bei dieser Einstellung also den gesamten Grundversionsspeicher aus (aber wer benötigt schon 512 verschiedene Zeichen?). Man nutzt diese Zeichenfülle also nur bei einem um mindestens 3 KByte erweiterten Speicher aus. In diesem Fall (und auch bei einer 8-KByte-Erweiterung) darf man den Bildschirmspeicher nicht außer acht lassen, denn dieser befindet sich ja auch im Bereich zwischen 4096 und 8192 (je nach Erweiterung). In beiden Fällen ergeben sich also Überschneidungen zwischen den beiden Speichern. So können bei einem 8-KByte-Speicher die unteren 512 Byte nicht für Zeicheninformationen verwendet werden (da dort ja der Bildschirmspeicher liegt). Also reduziert sich die Anzahl der verfügbaren Zeichen um 64.

Bei einer 3-KByte-Erweiterung wird man mit dem gleichen Problem konfrontiert. In diesem Fall werden die obersten 512 Bytes vom Videospeicher in Beschlag genommen.

ZG = 13 (Startadresse 5120): Mit diesen 384 Zeichen kommt man in der Regel gut aus (wie man alle auf den Bildschirm bringt, sehen wir in der nächsten Folge). Bezüglich der Nutzbarkeit der Zeichen ergibt sich nur bei der Grundversion beziehungsweise 3-KByte-Erweiterung das Problem, daß der obere Bereich (also Adresse 7680 bis 8192) vom Bildschirmspeicher belegt werden. Das betrifft die Bildschirmcodes 64 bis 128 im Klein-/Großschriftmodus.

ZG = 14 (Startadresse 6144): Hier haben wir Platz für 256 Zeichen, die bei 8 KByte voll nutzbar sind. Im anderen Fall sind wiederum 64 Zeichen für den Videospeicher zu subtrahieren.

ZG = 15 (Startadresse 7168): Diese Zeichenmenge wird üblicherweise in der Grundversion verwendet. Hierbei reduziert sich der Programmspeicher nämlich nur um 512 Byte (also 64 Zeichen). Die anderen 64 Zeichen fallen ja wiederum dem Bildschirmspeicher zum Opfer.

Da das Zeichenkopieren fast immer notwendig ist, verwendet man anstelle des Basicprogramms aus Geschwindigkeitsgründen besser eine Maschinenroutine (Listing 2 und 3). Das Programm kann mit Hilfe des Laders in ihre eigene Grafikrouti-

```
100 rem ********************************
110 rem *** dieses programm kopiert ***
120 rem *** zeichen aus dem rom ins ***
130 rem *** ram. die  routine  kann ***
140 rem *** in ihr eigenes grafik - ***
150 rem *** programm eingebaut wer- ***
160 rem *** den. lediglich die va - ***
170 rem *** riable  rg  muss  der   ***
180 rem *** routine  uebergeben     ***
190 rem *** werden.                 ***
200 rem ********************************
210 fort=0to48:readd:poket+828,d:s=s+d
220 next:rem *** daten einlesen
230 if s<>4890 thenprint"datenfehler !":
end
240 rg =14 : rem *** uebergabevariable
250 poke4,rg
260 sys828
270 data169,000,133,000,133
280 data002,169,128,133,001
290 data165,004,056,233,008
300 data010,010,133,003,166
310 data004,160,000,177,000
320 data145,002,230,000,230
330 data002,208,246,230,001
340 data230,003,202,208,239
350 data173,005,144,005,004
360 data141,005,144,096

ready.
```

Listing 2. Zeichensatz kopieren (Basic-Lader)

```
****************** character copy
033c  lda #$00    ; pointer init
033e  sta $00
0340  sta $02
0342  lda #$80    ; zeiger auf zeichen-
0344  sta $01     ; generator rom
0346  lda $04     ; rg variable holen
0348  sec
0349  sbc #$08    ; umrechnung in h-byte
034b  asl         ; multiplikation mit 4
034c  asl
034d  sta $03
034f  ldx $04     ; schleifenzaehler
0351  ldy #$00
0353  lda ($00),y; zeichen laden
0355  sta ($02),y; im ram abspeichern
0357  inc $00     ; low-byte inc.
0359  inc $02
035b  bne $0353   ; uebertrag ?
035d  inc $01     ; ja, dann high-byte
035f  inc $03     ; inkrementieren
0361  dex         ; zaehlschleife
0362  bne $0353
0364  lda $9005   ; auf ram-zeichen um-
0367  ora $04     ; schalten
0369  sta $9005
036c  rts
```

Listing 3. Zeichensatz kopieren (Assembler-Darstellung)

```
10 rg=14:zg=(rg-8)*1024
20 fort=0to8191-zg
30 pokezg+t,peek(32768+t)
40 next
50 poke36869,peek(36869)orrg

ready.
```

Listing 1. Zeichensatz ins RAM kopieren

Wert 1	Wert 2	Verknüpfung
0	0	0
0	1	1
1	0	1
1	1	1

Tabelle 4a. Die Wahrheitstabelle der ODER-Operation

Wert 1	Wert 2	Verknüpfung
0	0	0
0	1	0
1	0	0
1	1	1

Tabelle 4b. Die Wahrheitstabelle der UND-Operation

ne eingebaut werden. Auch speicherplatzmäßig ergeben sich, da das Maschinenprogramm recht kurz ist, keine Probleme, denn es liegt im Bandpuffer (Adresse 828).

Das Ändern der Sonderzeichen

Nun sind wir an einer Stelle angelangt, von der aus wir die Zeichen nach eigenen Wünschen abändern können, seien es mathematische Sonderzeichen, deutsche Umlaute oder Grafikzeichen für eigene Spielprogramme.

Am Anfang steht der Entwurf eines Zeichens mit Hilfe einer 8 x 8 Matrix (Bild 5). Hier in unserem Beispiel soll das Pfundzeichen £ (Bildschirmcode 28) durch das Copyrightzeichen ersetzt werden.

Dazu wird unser Beispielprogramm aus Listing 1 um folgende 5 Zeilen ergänzt:

```
60 CH = 28: AD = CH *8 + ZG:
REM Formel #1
70 FOR Z = 0 TO 7
80 READ D
90 POKE AD*Z,D: NEXT
100 DATA 60, 66, 153, 161, 153, 66,
60
```

Der Bildschirmcode des Zeichens wird durch die Variable CH übergeben. Daraus errechnet sich das Programm die Startadresse des Zeichens, indem es den Bildschirmcode mit 8 multipliziert und die Anfangsadresse des Zeichenspeichers dazuaddiert (in unserem Fall 6144).

Das Abspeichern der Zeichen

Im allgemeinen erstellt man solche Zeichen nicht aus Spaß an der Freude, sondern man möchte sie in Spielen oder anderen Programmen verwenden. Dazu müssen sie in irgendeiner Form abgespeichert werden. Hierfür gibt es wieder zweierlei Möglichkeiten. Entweder man schreibt die Zeicheninformationen in Form von DATA-Zeilen ins Programm oder man SAVEt sie direkt auf Band oder Diskette ab.

Zeile	7.	6.	5.	4.	3.	2.	1.	0.	Bit
0.	0	0	1	1	1	1	0	0	= 60
1.	0	1	0	0	0	0	1	0	= 66
2.	1	0	0	1	1	0	0	1	= 153
3.	1	0	1	0	0	0	0	1	= 161
4.	1	0	1	0	0	0	0	1	= 161
5.	1	0	0	1	1	0	0	1	= 153
6.	0	1	0	0	0	0	1	0	= 66
7.	0	0	1	1	1	1	0	0	= 60

Bild 5. Auch die eigenen Zeichen muß man mit Hilfe der Matrix entwerfen (hier als Beispiel das Copyright-Zeichen)

Beide Verfahrensweisen eignen sich für bestimmte Anwendungsgebiete besonders gut, für andere weniger gut.

Die DATA-Zeilen-Methode eignet sich dann, wenn es darum geht, lediglich 3 oder 4 Zeichen abzuändern (beispielsweise für ein Textverarbeitungsprogramm mit deutschen Umlauten). Diese werden dann — wie in dem Beispiel oben — als DATA-Zeilen ins Programm geschrieben.

Für größere Änderungen am Zeichenvorrat ist diese Methode aber aus Geschwindigkeits- und Platzgründen (sie benötigt etwa viermal so viel Platz) denkbar ungeeignet.

Bei der anderen Methode werden die Zeicheninformationen wie ein Programm einfach auf Band abgespeichert. Dazu besinnen wir uns wieder auf Folge 1 in Ausgabe 9, wo von den Basic-Zeigern die Rede war. Ferner ist dort beschrieben, wie man Platz für Maschinenprogramme und Sonderzeichen schafft, und sie vor dem Zugriff des Basicinterpreters schützt. Auch hier muß man — wie so oft, wenn es um Grafik oder Bildschirm geht — zwischen den zwei grundsätzlichen Ausbauversionen unterscheiden.

All diese Vorgänge möchte ich anhand von Bild 6 erklären (6a für die Grundversion, 6b für die 8-KByte-Erweiterung):

Grundversion (Bild 6a): Der dunkelgelb unterlegte Teil der Speichergrafik stellt den Adreßbereich dar, auf den der Interpreter zurückgreift. Dieser geht normalerweise bis Adresse 7680 (zu erkennen an der hellgelben Farbe).

Durch die Umstellung mit POKE 55,0: POKE 56, 28:CLR hat man Platz für 64 Zeichen (= 512 Byte) geschaffen, die vom Interpreter nicht angetastet werden (dies haben wir ja in Folge 1 schon besprochen).

Nun aber zu dem Abspeichern des Zeichensatzes. Auch beim SAVEn richtet sich der VC nach diesen Zeropagezeigern, denn er speichert alles ab, was er zwischen den beiden Adreßpaaren 43, 44 (Basic-Anfang) und 45,45 (Programmende) findet. Folglich gibt das erste Paar die Anfangs-, das andere die Endadresse der zu speichernden Daten an. Normalerweise sind

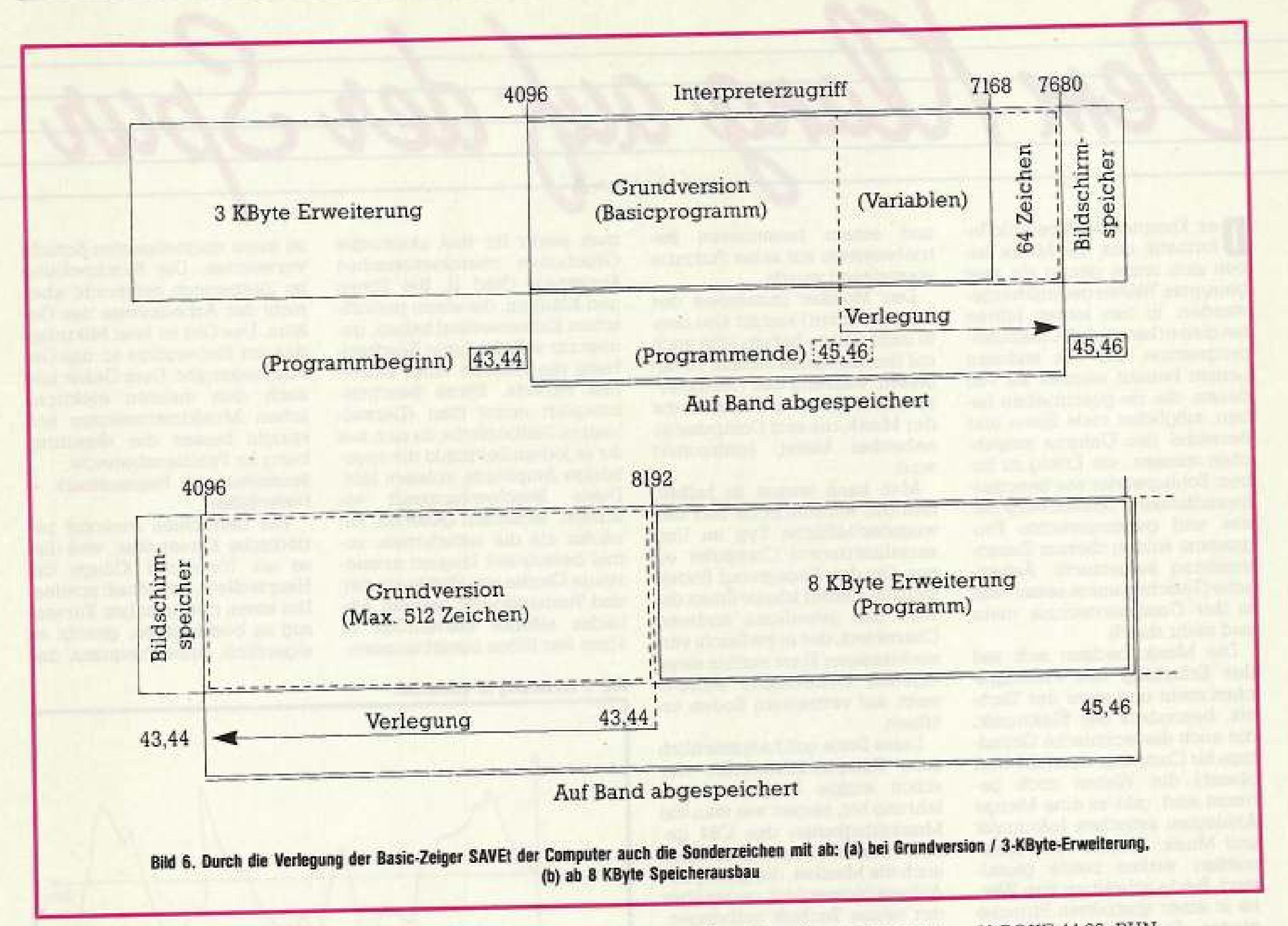


Bild 6. Durch die Verlegung der Basic-Zeiger SAVEt der Computer auch die Sonderzeichen mit ab: (a) bei Grundversion / 3-KByte-Erweiterung, (b) ab 8 KByte Speicherausbau

diese Daten das Basic-Programm, welches sich zwischen diesen Zeigern befindet. Durch eine Änderung der beiden Zeropagespeicherstellen 45 und 46 auf das Ende des Zeichensatzs (Adresse 7680) bewirkt man, daß das gesamte Programm mit Variablenbereich und Zeichensatz abgespeichert wird (Bild 6a — rote Markierung).

Schritt für Schritt auf Band

Hier noch einmal die nötigen Programmschritte:

1. Programm um Zeile 5 ergänzen.

5 POKE 45, XXX: POKE 46, XX: POKE 55, 0: POKE 56, 28: CLR

Warum muß das Programm um eine Zeile ergänzt werden? Nun, da sich der Computer bei der Variablenverwaltung nach diesen Zeigern richtet, müssen diese nach dem Ladevorgang wieder auf den alten Wert — der auf das wirkliche Programmende zeigt — gesetzt werden, damit der Computer nicht mit der Verwaltung durcheinander kommt. Auch die Zeichen selbst müssen nach dem Laden wieder vor dem Interpreterzugriff geschützt werden, welches durch die Veränderung der Zeiger 55, 56 geschieht.

2. Nun stellen wir quasi »zu Fuß« den Inhalt von Adresse 45 und 46 fest:

PRINT PEEK(45), PEEK(46)

3. Diese Werte werden nun nachträglich (statt »xxx«) in Zeile 5 geschrieben, wobei zu beachten ist, daß der Wert von Adresse 45 ein-, zwei- oder dreistellig sein kann (oben wurden drei Stellen angenommen). Sollten es nun nicht drei Ziffern sein, so ist der Rest mit Nullen zu ergänzen (zum Beispiel 2 = 002 oder 34 = 034). Wichtig ist, daß sich die Zeilenlänge nicht ändert, da sich damit auch das Programmende verschieben würde.

4. Die Zeiger werden nun zum Abspeichern vorbereitet: POKE 45, 0: POKE 46, 30: CLR: SAVE»...«.

Wie man sieht, befindet sich das Programm vor den Sonderzeichen im Speicher. Die Zeiger müssen zum Abspeichern also nach oben gesetzt werden. Bei einer 8-KByte-Erweiterung liegt der Fall genau anders herum. Dort liegt der Zeichensatz nämlich vor dem Programm. Folglich muß hier der Basic-Anfang beim Abspeichern nach unten gelegt werden, aber dies besprechen wir jetzt im Folgenden noch genauer:

8-KByte-Erweiterung (Bild 6b): Auch hier ist die erste Handlung das Verstellen eines Basic-Zeigers. Auffällig ist, daß die Sonderzeichen — wie oben bereits angesprochen — nicht mehr oberhalb des Basic-Programms (also wie in Bild 6a zu sehen an dessen Ende), sondern unterhalb liegen. Daher muß die Basic-Anfangsadresse so geändert werden, daß auch hier kein zerstörendes Eingreifen mehr möglich ist. Durch das Hochsetzen der Startadresse von 4608 auf 8129 wird dies erreicht: POKE 44, 32: POKE 8192,0: NEW

Diese Anweisung ist vor dem Laden oder der Eingabe des Programms notwendig, denn wie man sieht, muß nach der erstmaligen Umschaltung der Programmspeicher mit NEW gelöscht werden.

Nachdem sich nun das Programm und die Sonderzeichen im Speicher befinden, können beide zusammen wieder abgeSAVEt werden; auch dazu ein »Rezept«:

1. Den Inhalt von Zeigerpaar 45,46 feststellen und notieren.
2. Basic-Anfang auf 4608 (Ursprungswert) zurückstellen: POKE 44,18: NEW
3. Zeile eingeben:

10 POKE 44,32: RUN

4. Den notierten Zeigerinhalt in die beiden Zeropagespeicherstellen zurückschreiben: POKE 45,Low-Byte: POKE 46, High-Byte: CLR
5. Die Zeichen mit dem Programm abspeichern (SAVE).

Lädt man das Programm wieder in den Speicher, so muß das High-Byte des Basic-Anfangszeigers (44) auf 32 gestellt werden, denn dort befindet sich ja das eigentliche Programm. Zeile 10 in Schritt 3 hat diese Aufgabe. Die Zeile befindet sich am zurückgestellten Programmbeginn (4608), also noch vor dem Zeichensatz, der erst bei Adresse 5120 beginnt. Das hat den Vorteil, daß man das Programm direkt mit RUN starten kann.

Soweit der wichtige Abschnitt über das Abspeichern der Sonderzeichen, der mit Absicht etwas umfangreich ausgefallen ist, denn auch derjenige, der nicht so viele Kenntnisse über den VC 20 hat, soll in der Lage sein, seine grafischen Werke auf Band zu bringen.

Mit dieser Erkenntnis beschließen wir die Einführung in die Grafikfähigkeit des VC 20. Das nächste Mal benutzen wir die bisher gewonnenen Grundlagen, um voll in die Materie einzusteigen.

(Christoph Sauer/ev)

Dem Klang auf der Spur

Der Komplex Mathematik/Informatik und die Musik haben sich lange genug als zwei getrennte Welten gegenübergestanden. In den letzten Jahren hat man erkannt, daß Computerprogramme, die von anderen Leuten benutzt werden als von denen, die sie geschrieben haben, möglichst viele Sinne und Bereiche des Gehirns ansprechen müssen, um Erfolg zu haben. Schlagwörter wie Benutzerfreundlichkeit, Softwareergonomie und gehirngerechte Programme sind in diesem Zusammenhang aufgetaucht. Ästhetische Gesichtspunkte setzen sich in der Computertechnik mehr und mehr durch.

Die Musik bedient sich seit der Erfindung des Phonographen mehr und mehr der Technik, besonders der Elektronik, die auch die technische Grundlage für Computer darstellt. Und obwohl die Welten noch getrennt sind, gibt es eine Menge Analogien zwischen Informatik und Musik: Musiker und Informatiker wirken beide gestaltend. Beide schreiben ihre Werke in einer abstrakten Sprache nieder. Die Werke erwachen erst durch eine Interpretation zum Leben und vermögen uns dann zu faszinieren. Was dem Musiker sein Instrument ist, ist für den Informatiker der Computer. Sowohl in der Informatik als auch in der Musik gab und gibt es Tendenzen, universelle Instrumente, die möglichst alles können sollen, zu schaffen. In der Musik zeugen die gewaltigen Kirchenorgeln und die Orchestrions des vorigen Jahrhunderts von diesen Bemühungen. Programmgesteuerte Rechner waren schon bald nach ihrer Erfindung mehr als nur Rechenmaschinen. Die technische Annäherung der beiden Bereiche ist heute schon so weit fortgeschritten, daß, zumindest was elektronische Musik anbelangt, es keinen wesentlichen Unterschied mehr zwischen Computern und Musikinstrumenten gibt. Mit Computern kann man heute nicht nur Noten editieren und Partituren plotten lassen, sondern Musik direkt auf vielfältige Weise wiedergeben. Ein moderner Synthesizer kann heute nicht nur typisch »elektronische« Klänge erzeugen, sondern auch [illegible]bendige Naturklänge bis hin [illegible] menschlichen Stimme. Und [illegible]nauerer Betrachtung er[illegible] sich als ein Computer, [illegible]igen Sonderbauteilen und einem besonderen Betriebssystem auf seine Aufgabe spezialisiert wurde.

Der Musiker (zumindest der Studiomusiker) kommt also heute nicht darum herum, sich auch mit der Computertechnik zu befassen, während der Computerspezialist eher spielerisch mit der Musik, die sein Computer so nebenbei bietet, konfrontiert wird.

Man kann wagen zu hoffen, daß der künstlerische und der wissenschaftliche Typ im Universalinstrument Computer einen Ort der Begegnung finden werden. Dieser könnte ihnen die Welt des jeweiligen anderen Charakters, der in vielleicht vernachlässigter Form auch in einer eigenen Gehirnhälfte schlummert, auf vertrautem Boden eröffnen.

Diese Serie soll hauptsächlich dem Computeranwender, der schon einige Programmiererfahrung hat, zeigen wie man die Musikfähigkeiten des C64 gezielt einsetzen kann, doch sollen auch die Musiker, die genügend Aufgeschlossenheit gegenüber der neuen Technik mitbringen, nicht zu kurz kommen.

Die Grundlagen

Um die Wirkungsweise elektronischer Musikinstrumente zu verstehen und um Musik einer Bearbeitung mit dem Computer zugänglich zu machen, müssen wir akustische Ereignisse mit Mitteln der Physik und der Mathematik beschreiben. Wenn auch unsere Sprache eine sehr technische ist, werden wir doch so oft wie möglich den Bezug zur traditionellen Sprache der Musik herstellen. Da wir stilistisch zunächst keine experimentelle Musik machen wollen, sondern Musik im traditionellen Stil mit dem Computer realisieren wollen, ist dies sogar unumgänglich.

Musik nicht mehr als Schwankungen des Schalldrucks?

Auslösend für eine Wahrnehmung des Gehörs ist ein sich schnell verändernder Schalldruck. Treten diese Veränderungen periodisch, das heißt zeitlich regelmäßig auf, so nimmt das Gehör einen Ton oder einen Klang wahr, andernfalls ein Geräusch. Trägt man den sich verändernden Schalldruck nach oben gegen eine rechtsgerichtete Zeitachse auf, erhält man einen für das akustische Geschehen charakteristischen Kurvenzug (Bild 1). Bei Tönen und Klängen, die einen periodischen Kurvenverlauf haben, genügt zur vollständigen Beschreibung die Angabe einer einzelnen Periode. Diese Beschreibungsart nennt man »Darstellung im Zeitbereich«, da sich aus ihr zu jedem Zeitpunkt die zugehörige Amplitude ablesen läßt. Diese Beschreibungsart erscheint, technisch gesehen, zunächst als die natürlichste, zumal bereits seit langem existierende Geräte wie Plattenspieler und Tonbandgerät getreue Abbilder solcher Kurvenzüge in Form von Rillen beziehungsweise einer magnetisierten Schicht verwenden. Die Beschreibung im Zeitbereich entspricht aber nicht der Arbeitsweise des Gehörs. Das Ohr ist kein Mikrofon, das nur Kurvenzüge an das Gehirn weitergibt. Dem Gehör und auch den meisten elektronischen Musikinstrumenten entspricht besser die »Beschreibung im Frequenzbereich«.

Beschreibung im Frequenzbereich — Fourier-Reihen

Wir betrachten zunächst periodische Kurvenzüge, weil diese als Töne und Klänge die Hauptrolle in der Musik spielen. Um einen periodischen Kurvenzug zu beschreiben, genügt es eigentlich, seine Frequenz, das

Bild 1. Darstellung im Zeitbereich

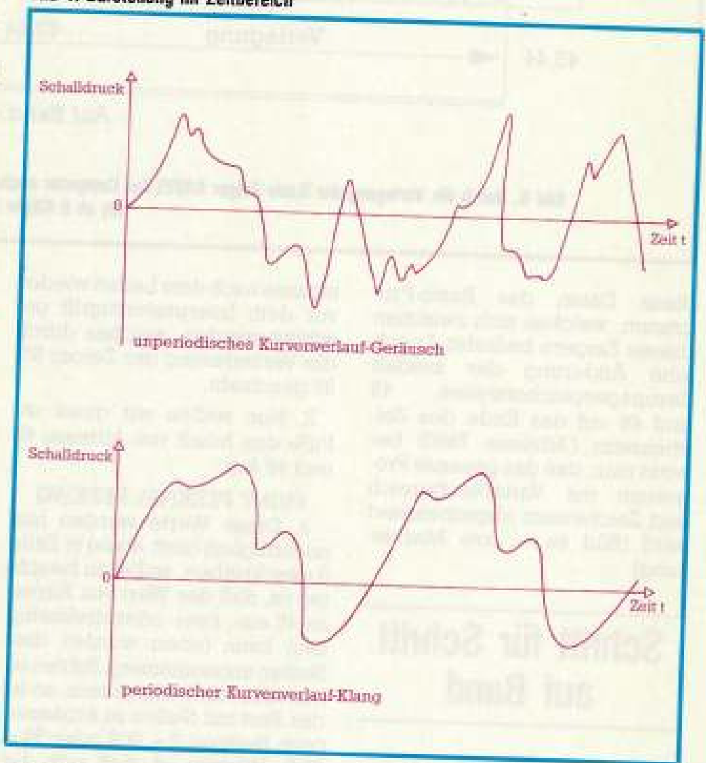
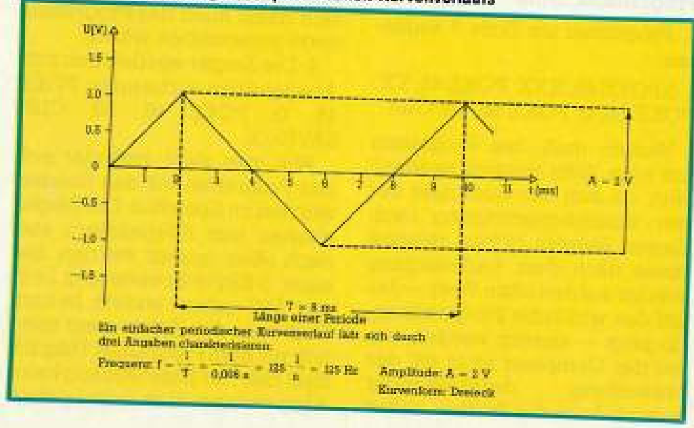


Bild 2. Charakterisierung eines periodischen Kurvenverlaufs



Teil 2

Bevor wir in der nächsten Folge genauer auf die Musik- und Klangerzeugung auf dem SID eingehen, soll in dieser Folge die Musik im allgemeinen behandelt werden.

heißt die Anzahl der Perioden, die pro Sekunde wiederkehren, seine Amplitude, also den Wert des maximalen Ausschlags des Kurvenzuges nach oben und nach unten und die Kurvenform zu kennen (Bild 2). Frequenz und Amplitude lassen sich leicht durch Zahlen ausdrücken. Da Frequenz und Amplitude für Tonhöhen- und Lautstärkeempfinden verantwortlich sind, hat man hier schon zwei musikalisch wichtige Parameter erfaßt. Die Kurvenform, die für den Klangcharakter verantwortlich ist, kann man allerdings nicht so einfach beschreiben. Es gibt aber einige einfache spezielle Kurvenformen, die sowohl theoretisch als auch musikalisch eine besondere Rolle spielen:

Die Sinus-, Rechteck-, Sägezahn- und Dreieck-Kurve

Diese Kurvenformen lassen sich mathematisch und grafisch einfach darstellen und auch technisch mit vertretbarem Aufwand mit einer Schaltung erzeugen. Der Sinus-Funktion kommt aber noch eine besondere Bedeutung zu. Akustisch wird der Sinuston, den kein natürliches Instrument erzeugen kann, als dumpf, undifferenziert und ohne ausgeprägten Charakter empfunden. Den Sinuston kann man sich als das einfachste mögliche akustische Ereignis vorstellen, aus dem man sich alle anderen Klänge mit periodischem Kurvenzug zusammengesetzt denken kann. Diese Betrachtungsweise läßt sich mathematisch durch die sogenannte »Entwicklung in Fourier-Reihen« rechtfertigen: Gehen wir von einer periodischen und stetigen Funktion aus. Stetig bedeutet anschaulich, daß der Kurvenzug keine Sprünge macht beziehungsweise, daß er sich in einem Zug zeichnen läßt ohne daß man absetzen muß. Die ideale Rechteck- und Sägezahnfunktion ist nicht stetig. Die Sprungstellen sind aber bei den realen elektro-akustischen Kurven nur mehr oder wenig steile Flanken. Eine solche Funktion kann man nun als Summe von geeigneten Sinus- und Cosinus-Funktionen schreiben (Bild 3).

Diese Summen bestehen im allgemeinen aus unendlich vielen Summanden. Eine solche Summe nennt man in der Mathematik eine Reihe. Die Fourier-Reihen bestehen aus Sinus- und Cosinus-Funktionen mit Frequenzen, die immer nur ganzzahlige Vielfache der Grundfrequenz betragen. Andere Frequenzen treten nicht auf. Die erste Komponente der Reihe nennt man auch Grundton, die weiteren Obertöne. An der unendlichen Anzahl der Summanden darf man sich nicht stoßen; eine solche Summe ist nämlich so zu verstehen: Betrachtet man nur eine Teilsumme, zum Beispiel mit allen Summanden bis zur fünffachen Grundfrequenz, so erhält man eine Funktion, die zwar nicht exakt die darzustellende Funktion ist, die sie aber bestmöglich annähert, so gut das mit fünf Sinus- und Cosinus-Funktionen eben möglich ist.

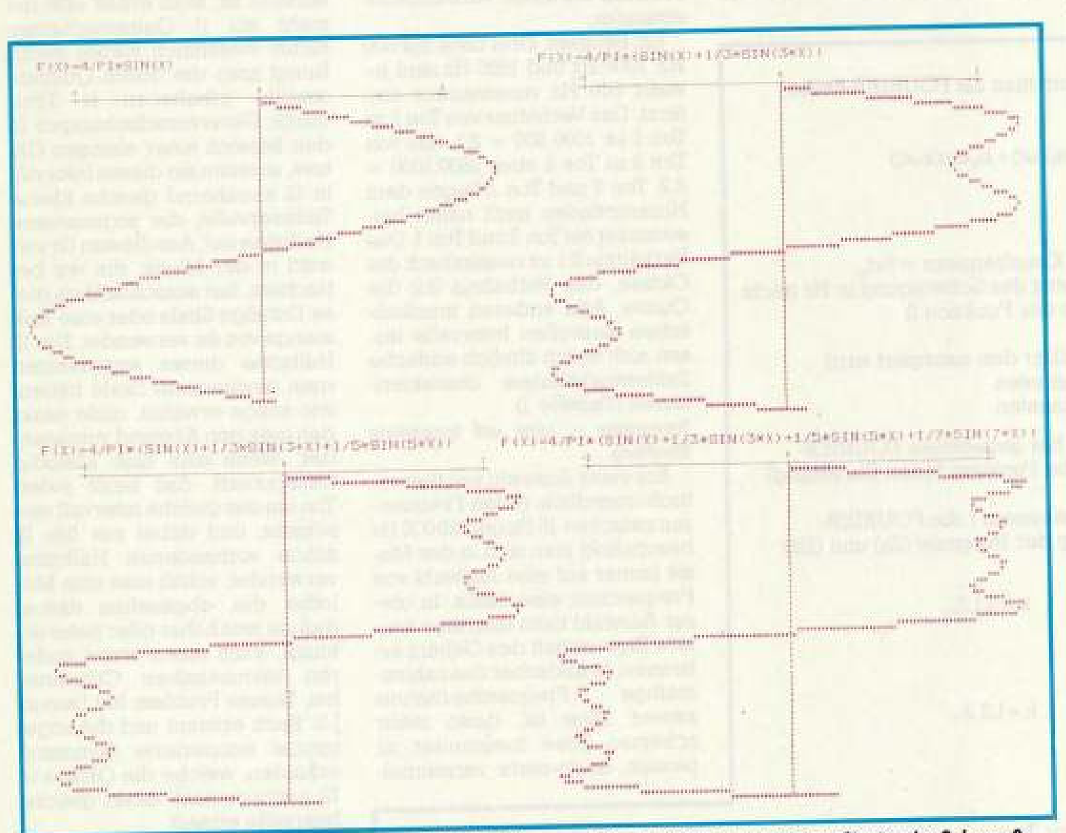


Bild 3. Annäherung einer Rechteckfunktion durch FOURIER-Teilsummen. Die Annäherung geht bis zum Oberton der Ordnung 9.

Nimmt man weitere Obertöne hinzu, zum Beispiel noch den sechsten und den siebten Oberton, dann wird die Annäherung besser. Es ist durch Hinzunahme von immer mehr Obertönen in die endliche Summe möglich, die Originalkurvenform beliebig genau anzunähern, wobei Obertöne mit wachsender Frequenz anteilmäßig beliebig klein werden. Nur unter diesen Umständen darf man in der Mathematik von einer unendlichen Summe (= Reihe) sprechen. Diese Umstände sind aber bei den Fourier-Reihen durch die Theorie gesichert.

Beschreibung im Frequenzbereich — das Spektrum

Eine Fourier-Reihe wird bereits allein durch die Angabe ihrer Koeffizienten vollständig beschrieben. Die Koeffizienten sind die Zahlen, die angeben, wie stark die einzelnen Grund- und Obertonanteile vertreten sind. Die Folge dieser Koeffizienten nennt man das Spektrum der dargestellten Kurvenform. Diese Koeffizienten beschreiben einen Klang genauso vollständig wie die der Kurve zugrundeliegende Funktion. Eine anschauliche Darstellung des Spektrums erhält man, wenn man auf einer waagrechten Achse in gleichmäßigen Abständen die Frequenzen des Grundtones und der Obertöne kennzeichnet und darauf Balken errichtet, deren Höhe den Koeffizientenwerten entspricht.

Die Fourier-Reihe beziehungsweise das Spektrum eines Klanges nennt man auch »Beschreibung im Frequenzbereich«. Die beiden vorgestellten Darstellungsarten von Klängen sind gleichwertige Beschreibungen ein und derselben Sache und lassen sich ineinander umrechnen (Bild 4).

Wie das Gehör arbeitet

Unter Gehör wollen wir das Ohr zusammen mit dem Teil des Gehirns verstehen, der für die Verarbeitung von akustischen Reizen zuständig ist. Dem Gehör wird nun die spektrale Sichtweise von Klängen weitaus besser gerecht als die Sichtweise als Kurvenformen. Der schallverarbeitende Teil des Innenohres (Schnecke mit Basilarmembran und dem Cortischen Organ) zerlegt in der Tat die eintreffenden Schwingungen in ihre spektralen Bestandteile und leitet diese einzeln über zirka 30000 Nervenfasern an das Gehirn weiter. Das Gehirn verarbeitet also Spektren und nicht Kurvenformen. Es macht einem geschulten Gehör keine Mühe, aus einem Orchester einzelne Instrumente herauszuhören. Musiker können sogar aus dem Klang eines einzelnen Instruments die Obertöne einzeln für sich hören. Sieht man sich die Kurvenform eines Orchesterklanges mit Mikrofon und Oszilloskop an, so sieht man nur Chaos, betrachtet man aber das Spektrum, so kann man durchaus einzelne Instrumente oder Instrumentengrup-

Dem Klang auf der Spur Teil 2

pen wiedererkennen. Die Leistungsfähigkeit der spektralen Betrachtungsweise wird hier deutlich. Das menschliche Gehör kann im Idealfall Frequenzen im Bereich von 16 Hz bis 20000 Hz wahrnehmen. Man kann daher bei Spektren Obertöne über 20000 Hz unberücksichtigt lassen. Ein Beispiel: Ein Sinuston und ein Sägezahnklang, beide von 400 Hz, klingen sehr unterschiedlich. Während der Sägezahn ein reichhaltiges Obertonspektrum besitzt, besteht der Sinus nur aus einem Grundton. Der Sägezahn klingt daher auch hell und scharf, während der Sinuston als weich und dumpf empfunden wird. Beträgt die Grundfrequenz der beiden Kurven aber zum Beispiel 16000 Hz, so kann man keinen Unterschied mehr hören, weil bereits der erste Oberton des Sägezahn mit 32000 Hz weit jenseits der Hörgrenze liegt.

Der Ton macht die Musik — über hohe und tiefe Töne

Wenn wir mit elektronischen Mitteln Musik machen wollen, müssen wir über Frequenzen genau Bescheid wissen. Das Gehör leitet die empfundene Tonhöhe grundsätzlich vom Grundton ab, unabhängig vom restlichen Spektrum, das für die Klangfarbe verantwortlich ist. Wir müssen uns zunächst also nur mit dem Grundton befassen. Eine interessante Eigenschaft des Gehörs ist, daß es absolute Tonhöhen schlecht, relative aber sehr gut bestimmen kann.

Spielt man einen beliebigen Ton aus der Stille heraus ohne jede Vergleichsmöglichkeit, so wird sich selbst ein Musiker in der Beurteilung der Tonhöhe um bis zu mehreren Halbtönen verschätzen (außer den wenigen, die mit einem »absoluten Gehör« gesegnet sind). Spielt man aber innerhalb eines Musikstücks nur einen Viertel- oder gar einen Achtelton falsch, so hört das sogar ein unmusikalischer Laie.

Das Gehör kann also Frequenzabstände gut beurteilen, arbeitet dabei aber logarithmisch. Das bedeutet, daß der empfundene Tonabstand, den man in der Musik als Intervall bezeichnet, nicht von der Differenz der Frequenzen abhängt, sondern von ihrem Verhältnis zueinander.

Ein Beispiel: Drei Töne mit 500 Hz, 1000 Hz und 1500 Hz sind jeweils 500 Hz voneinander entfernt. Das Verhältnis von Ton 2 zu Ton 1 ist 1000:500 = 2:1, das von Ton 3 zu Ton 2 aber 1500:1000 = 3:2. Ton 3 und Ton 2 liegen dem Hörempfinden nach näher beieinander als Ton 2 und Ton 1. Das Verhältnis 2:1 ist musikalisch die Oktave, das Verhältnis 3:2 die Quinte. Alle anderen musikalischen sinnvollen Intervalle lassen sich durch ähnlich einfache Zahlenverhältnisse charakterisieren (Tabelle 1).

Tonsysteme — reine und temperierte Stimmung

Aus einer Auswahl von theoretisch unendlich vielen Frequenzen zwischen 16 Hz und 20000 Hz beschränkt man sich in der Musik immer auf eine Auswahl von Frequenzen, eine Skala. In dieser Auswahl kann man eine weitere Eigenschaft des Gehörs erkennen: Je einfacher das zahlenmäßige Frequenzverhältnis zweier Töne ist, desto mehr scheinen Töne zueinander zu passen, desto mehr verschmelzen sie zu einem Klangbild. Aus diesem Grund spielen in der Musik die Intervalle mit den einfachsten Frequenzverhältnissen, die Oktave (2:1), die Quinte (3:2) und die Quarte (4:3) die zentrale Rolle. Töne im Oktavabstand werden musiktheoretisch nicht einmal als etwas wesentlich Verschiedenes angesehen. Die abendländischen Tonskalen werden alle dadurch gewonnen, daß man von einem festen Ton ausgehend in Quint-, Quart- und Oktavabständen mehr oder weniger neue Töne in die Skala einbezieht. Wenn man sich mit dieser Methode um 12 Quinten nach oben vom Grundton entfernt hat, erhält man einen Ton, der annähernd 7 Oktaven vom Grundton entfernt ist. Man erhält also mit mehr als 11 Quintenschritten nichts wesentlich Neues mehr. Bringt man die durch Quintenschritte erhaltenen 11 Töne durch Oktavverschiebungen in den Bereich einer einzigen Oktave, so teilen sie dieses Intervall in 12 annähernd gleiche kleine Teilintervalle, die sogenannten Halbtöne auf. Aus diesem Grund wird in der Musik, die wir betrachten, fast ausschließlich diese 12stufige Skala oder eine Teilmenge von ihr verwendet. Die 12 Halbtöne dieser sogenannten »rein gestimmten« Skala haben, wie schon erwähnt, nicht exakt den gleichen Abstand voneinander. Wenn man eine Melodie transponiert, das heißt jeden Ton um das gleiche Intervall verschiebt, und dabei nur die 12 schon vorhandenen Halbtöne verwendet, erhält man eine Melodie, die, abgesehen davon, daß sie jetzt höher oder tiefer erklingt, auch einen etwas anderen harmonischen Charakter hat. Dieses Problem hat bereits J.S. Bach erkannt und die sogenannte temperierte Stimmung erfunden, welche die Oktave in 12 mathematisch exakt gleiche Intervalle einteilt.

Die temperierte Stimmung wird heute fast ausschließlich verwendet. Die Berechnung der Frequenzen für ein elektronisches Musikinstrument ist einfach: Ausgehend vom eingestrichenen a, dessen Frequenz international auf 440 Hz festgelegt ist, erhält man die anderen Frequenzen der anderen Töne der Oktave durch wiederholte Multiplikation mit dem Halbtonabstand $\sqrt[12]{2}$. Die Frequenzen der Töne anderer Oktaven erhält man durch Multiplikation und Division mit Zweierpotenzen (Tabelle 2).

Eine periodische Funktion f kann man als FOURIER-Reihe schreiben:

$$(1)\quad f(t) = \frac{a_0}{2} + \sum_{K=1}^{\infty} (a_k \cos(k\omega t) + b_k \sin(k\omega t))$$

Dabei ist:

ω	die sogenannte Kreisfrequenz $= 2\pi f_0$
f_0	die Grundfrequenz der Schwingung in Hz (nicht verwechseln mit der Funktion f)
t	die Zeit
k	der Laufindex, über den summiert wird
$a_0, a_1, \ldots$	FOURIER-Koeffizienten
$b_1, b_2, \ldots$	FOURIER-Koeffizienten

Formel (1) beschreibt, wie man bei gegebenen FOURIER-Koeffizienten (dem Spektrum) die Funktion f (den Kurvenzug) erhält.

Umgekehrt erhält man aus gegebenem f die FOURIER-Koeffizienten durch Berechnung der Integrale (2a) und (2b):

$$(2a)\quad a_k = \frac{2}{T} \int_0^T f(t)\cos(k\omega t)\,dt \qquad k = 0,1,2\ldots$$

$$(2b)\quad a_k = \frac{2}{T} \int_0^T f(t)\sin(k\omega t)\,dt \qquad k = 1,2,3\ldots$$

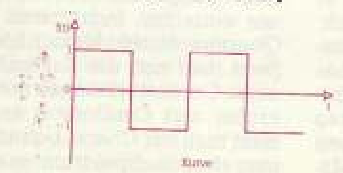
Dabei ist:

ω, t, k	wie oben
T	die Periodendauer $T = \frac{1}{f_0}$

Beispiel (ohne Rechnung) Rechteckfunktion

Zeitbereich

$$f(t) = \begin{cases} 1 \text{ für } t \in [2n\pi, (2n+1)\pi[\\ -1 \text{ für } t \in [(2n-1)\pi, 2n\pi[\end{cases}$$



Frequenzbereich

$$a_0 = a_1 = a_2 = \ldots = 0$$

$$b_1 = \frac{4}{\pi},\ b_3 = \frac{1}{3}\cdot\frac{4}{\pi},\ b_5 = \frac{1}{5}\cdot\frac{4}{\pi},\ b_7 = \frac{1}{7}\cdot\frac{4}{\pi}, \ldots$$

$$b_2 = b_4 = b_6 = \ldots = 0$$

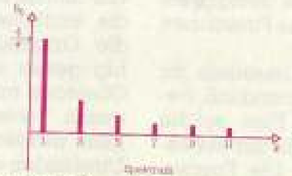


Bild 4. Korrespondenz zwischen Zeit- und Frequenzbereich

Intervall	Frequenzverhältnis
Oktave	2 : 1
große Septime	15 : 8
kleine Septime	16 : 9
große Sexte	5 : 3
kleine Sexte = übermäßige Quinte	8 : 5
Quinte	3 : 2
verminderte Quinte = übermäßige Quarte	10 : 7
Quarte	4 : 3
große Terz = verminderte Quarte	5 : 4
kleine Terz	6 : 5
große Sekund (Ganzton)	9 : 8
kleine Sekund (Halbton)	16 : 15

Tabelle 1. Frequenzverhältnisse der Intervalle

Es gibt 7 mit lateinischen Buchstaben benannte Stammtöne, die sich in allen Oktaven wiederholen. Die Nachsilbe 'is' bedeutet eine Erhöhung um einen Halbton, die Nachsilbe 'es' bedeutet eine Erniedrigung um einen Halbton.

Groß- und Kleinschreibung der Töne und hoch- und tiefgestellte Indizes kennzeichnen die Zugehörigkeit zu verschiedenen Oktaven.

Oktave	Note	Frequenz in Hz
Subkontraoktave	C_2 bis H_2	
Kontraoktave	C_1 bis H_1	
Große Oktave	C	65,4064
	Cis	69,2957
	D	73,4162
	Dis	77,7817
	E	82,4069
	F	87,3071
	Fis	92,4986
	G	97,9989
	Gis	103,8262
	A	110,0000
	Ais	116,5409
	H	123,4708
Kleine Oktave	c	130,8128
	cis	138,5913
	d	146,8324
	dis	155,5635
	e	164,8138
	f	174,6141
	fis	184,9972
	g	195,9977
	gis	207,6524
	a	220,0000
	ais	233,0819
	h	246,9417
Eingestrichene Oktave	c^1	261,6256
	cis^1	277,1826
	d^1	293,6648
	dis^1	311,1270
	e^1	329,6276
	f^1	349,2282
	fis^1	369,9944
	g^1	391,9954
	gis^1	415,3047
	a^1	440,0000
	ais^1	466,1638
	h^1	493,8833
Zweigestrichene Oktave	c^2	523,2511
	cis^2	554,3653
	d^2	587,3295
	dis	622,2540
	e^2	659,2551
	f^2	698,4565
	fis^2	739,9888
	g^2	783,9909
	gis^2	830,6094
	a^2	880,0000
	ais^2	932,3275
	h^2	987,7666

Tabelle 2. Frequenzen bei temperierter Stimmung und die Namen der Töne

Bevor wir detailliert auf die Klangerzeugung im C 64 eingehen, wollen wir uns erst einen Überblick über die wichtigsten Synthesetechniken in elektronischen Instrumenten verschaffen.

Additive und subtraktive Synthese in Orgeln

Der Unterschied zwischen Orgel und Synthesizer verwischt sich durch den zunehmenden Einzug der Digitaltechnik mehr und mehr. Hier soll aber noch einmal der Unterschied zwischen einer typischen Orgel und einem typischen Synthesizer, wie er noch vor einigen Jahren bestanden hat, dargestellt werden.

Synthesetechniken

In einer Orgel werden für jede einzelne Taste ein oder mehrere Töne zur Verfügung gestellt. Man braucht dazu so viele einzelne Tongeneratoren, wie das Instrument Tasten hat. Diese Tongeneratoren, die Sinus-, Rechteck- oder Sägezahnschwingungen erzeugen können, müssen alle einzeln gestimmt werden. Den Aufwand an Generatoren kann man reduzieren, wenn man nur die 12 Töne der obersten Oktave erzeugt und die Töne der weiteren Oktaven durch Frequenzteilung realisiert. Die hier erforderliche Teilung durch Zweierpotenzen ist schaltungstechnisch einfach zu lösen. Man kann sogar die 12 Töne der obersten Oktave durch Teilung aus einer noch viel höheren Master-Frequenz (im MHz-Bereich) gewinnen. Dadurch wird die Orgel stimmstabil und kann als Gesamtheit, nur durch Verändern der Masterfrequenz anderen Instrumenten angepaßt werden. Ein Nachteil dieses Teilerkonzepts ist, daß die Oktaven zu genau sind. Da Töne im Oktavabstand phasenstarr miteinander gekoppelt sind, klingen zwei solche Töne wie nur ein Ton, nur mit einem etwas volleren Oberton-Spektrum.

Charakteristisch für eine Orgel ist, daß alle Töne gleichzeitig und durchgehend bereitstehen und durch Tastendruck auf einen Verstärker durchgeschaltet werden, der sie hörbar macht. Dieses Konzept hat den Vorteil, daß beliebig viele Töne polyphon, das heißt gleichzeitig gespielt werden können. Gehen wir einmal davon aus, daß eine Generatorgruppe für alle Tasten Sinustöne zur Verfügung stellt. Dann findet man zumindest für die tieferen Töne der Klaviatur neben dem Grundton eine ganze Reihe von geeigneten Obertönen aus der Generatorgruppe. Diese kann man durch geeignete Verschaltung den Grundtönen zumischen, wobei der Anteil der Obertöne einzeln und stufenlos durch sogenannte Zugriegel eingestellt werden kann. Benötigt man auch für die hohen Töne der Klaviatur noch Obertöne, so muß die Generatorgruppe mehr Töne erzeugen können, als die Orgel Tasten hat. Einen solchen Klangaufbau aus einzelnen Obertönen nennt man »Additive Synthese«.

Filter

Nun kann die Generatorgruppe einer Orgel oft auch Rechteck- oder Sägezahnschwingungen erzeugen, die von sich aus schon sehr obertonreich sind. Leitet man diese Schwingungen durch Filter, wird der Obertongehalt abgewandelt: es entstehen weitere Klangfarben. Ein Filter ist eine Baugruppe, die Signale abhängig von ihrer Frequenz verstärkt oder abschwächt (Bild 5). Ein Filter macht aus einem Sinuston immer wieder einen Sinuston, nur mit veränderter Amplitude. Andere Signale werden durch ein Filter auch in ihrer Kurvenform geändert. Man kann sich in Gedanken vorstellen, daß ein Filter ein Signal in seine Spektralanteile zerlegt, jeden einzelnen sinusförmigen Anteil frequenzabhängig verstärkt oder abschwächt und schließlich diese Anteile wieder zu einer neuen Kurvenform zusammenbaut. Da Filter keine neuen Obertöne produzieren, sondern nur das Verhältnis von vorhandenen Obertönen zueinander verändern, spricht man von »subtraktiver Synthese«.

Perkussion, Tremolo, Vibrato

Eine Orgel, wie sie bis hierher beschrieben wurde, klingt noch sehr starr und unlebendig. Wenn auch additive Synthese und Filterung eine gewisse Variation der Klangfarbe zulassen, so ist der Klang trotzdem so unlebendig wie der einer Autohupe, die einfach ein- und ausgeschaltet wird. Da der zeitliche Lautstärkeverlauf viel vom Charakter eines Klanges ausmacht, kommt man nicht umhin, die Amplitude eines Klanges vom Anschlag der Taste bis zum Loslassen dynamisch zu beeinflussen. Ein Verstärker, dessen Verstärkung im Moment des Anschlags am größten ist und die dann exponentiell abklingt, sorgt für einen natürlichen, perkussiven Amplitudenverlauf. Den zeitlichen Verlauf der Amplitude eines Klanges nennt man seine

Hüllkurve.

Man kann bei Orgeln auch nur einzelne Obertöne abklingen und andere unverändert stehen lassen. Man erhält damit einen Klang, dessen Färbung sich im

zeitlichen Ablauf ändert. Man spricht dann von **Klangfarbendynamik**.

Mit **Tremolo** beziehungsweise **Vibrato** bezeichnet man langsame periodische Änderungen in der Lautstärke beziehungsweise in der Tonhöhe eines Klanges. Beide Effekte lassen den Klang voller, natürlicher und wärmer wirken. Ein gutes Vibrato erhält man mit rotierenden Lautsprechern. Dieser Effekt läßt sich aber auch mit rein elektronischen Mitteln realisieren.

Das klassische Synthesizer-Konzept von Moog

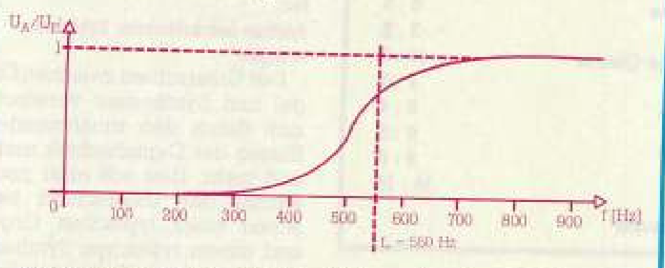
Eine Orgel, wie sie hier vorgestellt wurde, erzeugt trotz aller Feinheiten nur Klänge, die durch den Umfang der Generatorgruppe, durch fest eingestellte Filter und durch die verfügbaren Effektschaltungen für Perkussion, Vibrato und Tremolo festgelegt sind.

Der (klassische) Synthesizer dagegen ist ein Instrument, das keine festen Klangregister wie die Orgel besitzt. Er setzt sich aus Baugruppen zusammen, die unabhängig voneinander die verschiedenen Qualitäten eines Klanges bestimmen, wie Tonhöhe, Amplitude und Klangfarbe. Diese Baugruppen lassen sich dabei noch dynamisch beeinflussen. Bild 6 zeigt ein Schema eines einfachen klassischen Synthesizers. Die Funktionsblöcke waren früher physikalisch als Einzelmodule gebaut, die sich erst der Anwender zu seinem persönlichen Synthesizer zusammengestellt hat. Auch die Verbindungen der Blöcke untereinander lagen vollkommen in der Hand des Musikers. Das Schema in Bild 6 stellt nur eine Minimalkonfiguration mit Standardverkabelung dar.

Das Manual (Keyboard) schaltet nicht fertige Klänge durch wie bei der Orgel, sondern erzeugt lediglich eine Steuerspannung (Control Voltage CV), die der Tonhöhe entspricht, sowie ein digitales Signal (Gate), das anzeigt, ob eine Taste gedrückt ist oder nicht. Standard bei der Steuerspannung ist 1 V pro Oktave beziehungsweise $\frac{1}{12}$ V pro Halbton.

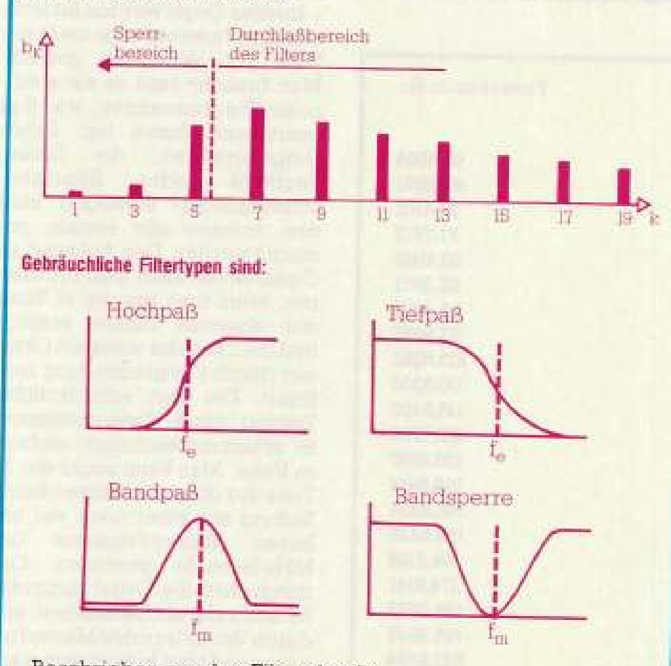
Die Steuerspannung steuert das »Herzstück« des Synthesizers, den spannungsgesteuerten Oszillator (Voltage Controlled Oscillator VCO), der die gängigen Kurvenformen wie Sinus, Dreieck, Sägezahn, Rechteck erzeugen kann. Er hat im allgemeinen eine exponentielle Steuercharakteristik von 1 V pro Oktave, damit er zum Keyboard paßt. Der VCO hat, wie auch die anderen spannungsgesteuerten Baugruppen, einen Steuerspannungs-Addierer. Damit kann man seine Frequenz über mehrere Eingänge gleichzeitig manipulieren.

Ein Filter verändert das Verhältnis der Obertöne eines Klanges zueinander. Filter werden durch ihren Frequenzgang beschrieben. Der Frequenzgang ist die Funktion, die das Verhältnis von Ausgangs- zu Eingangsamplitude bei Sinustönen, abhängig von der Frequenz, beschreibt.



Frequenzgang eines Hochpaßfilters mit einer Eckfrequenz (das ist die Frequenz, bei der die Filterwirkung von Sperren nach Durchlassen übergeht) von zirka 550 Hz.

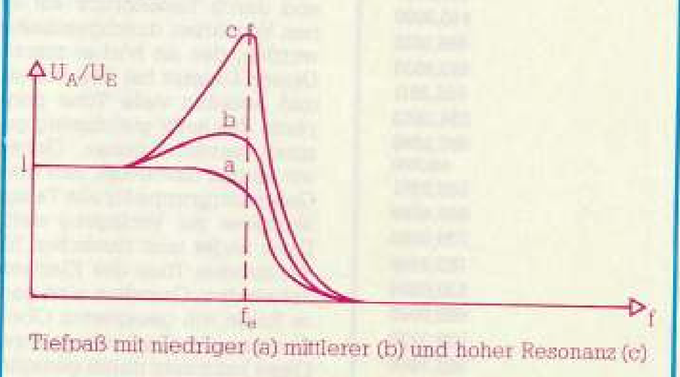
Geht man davon aus, daß das Rechtecksignal eine Grundfrequenz von 100 Hz besitzt, dann sieht das Spektrum nach Durchlaufen des Filters etwa so aus:



Beschrieben werden Filter durch:
Eckfrequenz f_e (Hoch- und Tiefpaß)
Mittelfrequenz f_m (Bandpaß und Bandsperre)
Steilheit: Das ist der Grad der Steigung des Frequenzgangs beim Übergang zwischen sperrendem und durchlässigem Zustand.
Resonanz siehe Zeichnung

Filterresonanz

Unter Resonanz versteht man die Verstärkung der Frequenzen um die Eck- beziehungsweise Mittenfrequenz. Alle bisher dargestellten Filter haben sehr niedrige Resonanz, da sie bei f_e beziehungsweise f_m nicht verstärken sondern nur mehr oder weniger abschwächen.



Tiefpaß mit niedriger (a) mittlerer (b) und hoher Resonanz (c)

Bild 5. Das Prinzip der Filter, die verschiedenen Filtertypen und die Filterresonanz

Die Schwingungen des VCO gelangen an ein spannungsgesteuertes Filter (Voltage Controlled Filter VCF). Man kann einstellen, ob sich das Filter wie ein Hochpaß, ein Tiefpaß, ein Bandpaß oder wie eine Bandsperre verhalten soll. Die Filterresonanz ist ebenfalls einstellbar. Der wichtigste Filterparameter, die Eckfrequenz, ist darüber hinaus auch durch eine Spannung steuerbar. Schließt man wie im Bild 6 die CV des Keyboards an das Filter mit an, so folgt es mit seiner Eckfrequenz exakt der Frequenz des zu filternden Klanges, was ein Festfilter in einer Orgel nicht kann.

Ein spannungsgesteuerter Verstärker (Voltage Controlled Amplifier VCA) sorgt schließlich für eine Amplituden-Hüllkurve. Die Steuerspannung stammt von einem Hüllkurvengenerator (Envelope Generator EG), im Bild 6 EG 2. Dieser erzeugt, ausgelöst durch den Gate-Impuls der Tastatur, einen der gewünschten Hüllkurve entsprechenden Spannungsverlauf. Es ist Standard geworden, Hüllkurven nach dem ADSR-Schema mit nur vier Parametern zu charakterisieren:

Attack — Anstiegszeit von Null auf Maximalpegel
Decay — Abklingzeit bei gedrückter Taste auf den Sustain-Pegel
Sustain — Pegel, der sich nach der Attack- und Decay-Phase einstellt
Release — Ausklingzeit nach Loslassen der Taste

Die Zeiten (A, D und R) sind im allgemeinen im Bereich von Millisekunden bis zu mehreren Sekunden einstellbar. Der Sustainpegel kann von Null bis zum Maximalpegel variiert werden.

Bei manchen Synthesizern ermöglicht ein weiterer EG (im Bild 6 EG 1) eine getrennte Beeinflussung des Filters. Damit ist eine Gestaltung der Klangfarbendynamik in weiten Grenzen möglich.

Auch Vibrato und Tremolo werden über eine weitere Steuerspannung bewirkt. Ein eigener Oszillator für niedrige Frequenzen von zirka 0,1—20 Hz (Low Frequency Oszillator LFO) ist dafür vorgesehen. Schaltet man den LFO auf den VCO so erhält man ein Vibrato, schaltet man ihn auf den VCA, so erhält man ein Tremolo. Die Tiefe der LFO-Modulation ist an jeder Baugruppe stufenlos einstellbar. Eine bei natürlichen Instrumenten nicht mögliche Modulation erhält man durch Beeinflussung des Filters durch den LFO.

Der Rausch-Generator (Noise-Generator) ist ein wichtiges Effekt-Element jedes Synthesizers. Rauschen wird durch modulierbare Filterung erstaunlich vielseitig und lebendig. Im

Dem Klang auf der Spur Teil 2

Spektrum des theoretisch idealen Rauschens sind alle Frequenzen gleichermaßen vertreten. Einzelne Grund- und Obertöne treten nicht auf. Da Rauschen kein periodisches Signal ist, kann man keine FOURIER-Reihe dafür angeben. Daß man dennoch von einem Spektrum reden darf, sichert eine erweiterte Theorie, auf die hier nicht eingegangen werden soll. Durch Rauschen mit dynamischer Filterung und Amplitudenhüllkurve lassen sich fast alle geräuschhaften Ereignisse nachbilden. Dies kommt noch mehr den Videospielen als der Musik zugute.

Unser Synthesizer-Schema kann man durch weitere Baugruppen ergänzen. So ergeben mehrere parallelgeschaltete VCOs einen volleren Klang, wenn man sie parallel stimmt, oder ermöglichen das Spielen von Akkorden, wenn man sie im Abstand musikalischer Intervalle stimmt. Mehrere LFOs gestatten reichhaltigere Möglichkeiten zu Modulationen.

Ein Nachteil soll aber nicht verschwiegen werden: Es kann immer nur ein Ton oder Klang gespielt werden. Möchte man polyphon spielen, so benötigt man einen Synthesizer, der aus so vielen Einzelschaltungen nach Bild 6 besteht, wie man maximal Töne gleichzeitig spielen möchte. Weiterhin benötigt man ein Keyboard, das mehrere unabhängige CV-Gate-Signal-Paare erzeugen kann. Der Schaltungsaufwand dafür ist sehr hoch. Man hat ihn aber heute dank fortschreitender Integration im Griff. So enthält der Sound-Chip des C 64 immerhin einen Synthesizer, der fast dem dreifachen des Standardschemas entspricht.

Digitaltechnik bei Analog-Synthesizern

Die Stärke des (Moog-)Synthesizers ist gerade, daß alle wesentlichen Eigenschaften der Module in weiten Grenzen einstellbar sind. Um aber eine einmal gefundene Einstellung reproduzierbar zu machen, muß man alle wichtigen Einstellparameter speichern und bei Bedarf diese gespeicherten Werte an den Modulen neu einstellen. Bei den ersten Synthesizern mußte man sich alle Werte aufschreiben und zur Reproduktion mühsam oft über hundert Potentiometern einstellen. Heute übernimmt diese Aufgabe weitgehend die Digitaltechnik. Es gibt kaum mehr professionelle Analog-Synthesizer, die nicht auch einen Mikrocomputer enthalten. Analog bedeutet hier, daß die Klangerzeugung weiter nach dem Schema von Bild 6 abläuft, das heißt, daß insbesondere die dynamische Steuerung der Module weiter über Steuerspannung erfolgt, daß aber die Einstellung von festen Parametern ein Mikrocomputer übernimmt. Dieser speichert ganze Parametersätze für verschiedene Sounds in nichtflüchtigen Speichern und belegt die Module auf Knopdruck mit einem gewünschten Parametersatz. Der Mikrocomputer kann auch ein polyphones Keyboard verwalten.

Bei weiterem Fortschreiten der Digitalisierung gibt es in einem Synthesizer keine Steuerspannungen mehr. VCO, VCF und VCA werden dynamisch mit Digitalwerten gesteuert. Sinngemäß spricht man dann auch von DCO, DCF und DCA. Die Funktion von LFOs und von Hüllkurvengeneratoren kann direkt von einem Mikrocomputer wahrgenommen werden. Dieser muß Folgen von digitalen Werten zur Verfügung stellen, die in ihrem zeitlichen Verlauf einer Modulationskurve oder einer Hüllkurve entsprechen. Theoretisch sind so zum Beispiel beliebige Hüllkurven denkbar, man hält aber meistens an dem bewährten ADSR-Schema fest.

Der Sound-Chip im C 64 entspricht diesem Konzept. Funktionell entspricht er ungefähr dreimal dem Schema nach Bild 6. Die Funktionsblöcke werden nicht mit Spannungen oder über Drehknöpfe gesteuert, sondern über Digitaldaten, die in eigenen Speicherplätzen des Synthesizers, den SID-Registern abgelegt werden müssen. Durch Adreß-Decodierungs-Hardware werden diese Register auf den CPU-Speicherbereich $D400 — $D41C abgebildet. Der Computer kann allein durch Belegung dieser Register mit sinnvollen Werten den Synthesizer-Chip steuern. Was dem C 64 zu einem vollständigen Sythesizer fehlt, ist ein Keyboard als musikergerechte Schnittstelle sowie festeingebaute Software zur Steuerung des Sound-Chips, denn mit dem Sound-Chip allein kann man noch keine Musik machen.

Das fehlende Keyboard ist nur dann ein Mangel, wenn man live auf dem C 64 spielen will. Wir werden in dieser Reihe sehen, wie man über das alphanumerische Tastenfeld spielen kann. Das soll aber nur experimentellen Charakter haben, zumal bereits ein brauchbares Musiker-Keyboard samt Software auf dem Markt ist. Die Stärke der Wiedergabe vorprogrammierter Musikstücke und in der Erzeugung von Sound-Effekten für Spiele. Daß es sich bei den Programmiertechniken, die dazu noch vorgestellt werden, nicht nur um reine Spielerei handelt zeigt die Tatsache, daß mit denselben Techniken auch zeitexaktes Steuern von mehreren professionellen Synthesizern über eine MIDI-Schnittstelle möglich ist.

Weitere interessante neuere Synthesetechniken sollen nur erwähnt, aber nicht näher beschrieben werden, da sie mit dem C 64 allein nicht zu verwirklichen sind.

Sound-Sampling bedeutet die vollständige Digitalisierung eines ganzen Kurvenverlaufs. Der Kurvenverlauf wird durch Abtastung natürlicher Schallereignisse gewonnen, oder ist das Ergebnis eines Rechenprozesses oder eine Kombination von beiden. Er befindet sich digitalisiert in einem Speicher, der periodisch ausgelesen wird, vergleichbar dem Auslesen des

Fortsetzung auf Seite 163

Bild 6. Schema eines klassischen Synthesizers, wie er auch im SID in dreifacher Ausführung vorkommt. Darunter die ADSR-Hüllkurve

Assembler ist keine Alchimie (Teil 5)

In dieser Folge des Assembler-Kurses wird die relative Adressierung erklärt. Damit verbunden sind auch die wichtigen Vergleichsoperationen. Anhand einer sehr häufig verwendeten Betriebssystem-Routine können Sie Ihr neu erworbenes Wissen testen.

In der letzten Ausgabe haben wir die Branch-Befehle kennengelernt. Heute wollen wir uns mit der relativen Adressierung dieser Befehle und noch einer anderen Art der Adressierung befassen. Weiterhin werden Sie einige neue Assembler-Worte lernen, nämlich die Vergleichsbefehle. Wie ganze Zahlen im Computer gespeichert sind, wissen wir bereits. Heute untersuchen wir die Speicherung von Zeichen. Schließlich werden wir unsere Nase noch ein wenig in die eingebaute Software des C 64 stecken.

Die relative Adressierung

Als wir den BNE-Befehl das erstemal verwendet haben, stellten wir fest, daß zum Beispiel BNE 1200 nicht — wie eigentlich zu erwarten war — ein 3-Byte-Befehl, sondern ein 2-Byte-Befehl ist. Damals mußten wir uns mit der Bemerkung zufrieden geben, es läge an der besonderen Art der Adressierung, nämlich der relativen Adressierung. Relativ bedeutet ja »bezogen auf etwas«. Wenn wir also beispielsweise BNE 1200 schreiben, liegt es nur an der Benutzerfreundlichkeit des SMON und vieler anderer Assembler, daß dieser die so geschriebene absolute Adresse 1200 in die richtige Form, nämlich die relative umrechnet. In Wahrheit verlangt der 6502 (und natürlich ebenso der 6510) eine Angabe darüber, wieviele Bytes nach vorne oder hinten im Programm er zur weiteren Programmverarbeitung springen (verzweigen) soll. Es gilt nun also, zwei Fragen zu klären:

1. Relativ wozu wird gesprungen und
2. Wie berechnet sich die Angabe, um wieviele Bytes nach vorne oder hinten im Programm der Sprung vollzogen werden soll.

Zur Klärung verwenden wir ein hypothetisches Programmsegment mit einem Sprungbefehl und sehen uns das Disassembler-Listing an:

```
2000 AD 00 30    LDA 3000
2003 F0 05       BEQ 200A
2005 A9 00       LDA #00
2007 8D 00 30    STA 3000
200A 60          RTS
```

Dieses Programm-Teilchen lädt den Inhalt der Speicherstelle 3000 in den Akku, überprüft dann, ob dieser Inhalt null ist und verzweigt beim Vorliegen der Null zum Rücksprung (RTS). Ist der Inhalt von 3000 nicht Null, dann wird 3000 auf Null gesetzt. 3000 könnte zum Beispiel eine Flagge sein.

Der Pfad, dem der Computer bei der Abarbeitung des Programmes folgt, wird durch den Programmzähler vorbereitet. Dieser ist dann, wenn der BEQ-Befehl an der Reihe ist, schon einen Schritt weiter, nämlich im Programmzähler steht dann die Adresse 2005.

Relativ zu dieser Adresse hat dann der Sprung zu erfolgen. Zum Inhalt des Programmzählers muß also die Sprungweite (auch häufig Offset genannt) addiert werden. Soweit zur Frage 1.

Zur Klärung von Frage 2 listen wir uns mal Byte für Byte unser Programm auf:

Byte		Inhalt	Bedeutung
2000		AD	LDA
2001		00	LSB von 3000
2002		30	MSB von 3000
2003		F0	BEQ
2004		05	Offset
2005		A9	LDA #
2006	1	00	
2007	2	8D	STA
2008	3	00	LSB von 3000
2009	4	30	MSB von 3000
200A	5	60	RTS

Neben der Byte-Nummer ist noch die Entfernung zu 2005 geschrieben. Daraus ist deutlich zu erkennen, daß die Sprungweite, die zum Programmzähler addiert wird, 05 sein muß, wenn der Sprung zum RTS erfolgen soll. Für Vorwärts-Verzweigungen gilt also: Von der Adresse des Befehls an, der auf den Branch-Befehl folgt, zählt man die Byte-Anzahl bis zum Sprungziel. Das Ergebnis ist der Offset.

Nun gibt es genauso häufig Rückwärts-Sprünge. In den bisher gezeigten Programmen sind sie mehrmals aufgetreten. Wie berechnet man den Offset in diesen Fällen? Sehen wir uns wieder das Disassembler-Listing eines solchen Programmsegmentes an:

```
1000    A2 00    LDX #00
1002    E8       INX
1003    D0FD     BNE 1002
1005    00       BRK
...
```

Dieses Programmchen tut nichts anderes, als das vorher auf Null gesetzte X-Register hochzuzählen, bis es über 255 läuft (dann tritt ja wieder 0 auf!). Solange der Inhalt des X-Registers ungleich Null ist, erfolgt ein Sprung zurück bis zur INX-Anweisung in Zeile 1002. Erst wenn die Null durch den Überlauf aufgetreten ist, endet das Programm mit einem BRK in Zeile 1005.

Wir wissen schon, daß der Programmzähler beim Verarbeiten des BNE-Befehls auf 1005 steht. Sehen wir uns auch dieses Programm Byte für Byte an:

Byte		Inhalt	Bedeutung
1000		A2	LDX #
1001		00	
1002	3	E8	INX
1003	2	D0	BNE
1004	1	FD	Offset
1005		00	BRK

Wieder ist neben der Bytenummer die Entfernung vom aktuellen Programmzählerstand angegeben. Wir müssen also vom Inhalt des Programmzählers 3 abziehen, um zum INX-Befehl in Byte 1002 zu gelangen. Das kennen wir aber schon aus den vergangenen Ausgaben: Wenn der Computer eine Zahl abzieht, dann addiert er das Zweierkomplement dieser Zahl. Hier soll nun 3 subtrahiert werden. Wir berechnen das Zweierkomplement:

3 = 0000 0011 (binär)

Das Einerkomplement davon ist:

1111 1100

Dann wird eine 1 addiert

1111 1101

Dies ist das Zweierkomplement. In hexadezimal ausgedrückt heißt diese Zahl $FD und ist unser Offset. Für Rückwärts-Verzweigungen gilt also: Von der auf die Branch-Anweisung folgenden Speicherstelle an zählt man die Bytes zurück bis zum Sprungziel. Das Zweierkomplement der sich dadurch ergebenden Byte-Anzahl ist der Offset.

Das sieht reichlich kompliziert aus, aber zum einen haben Sie ja einen ganz freundlichen Assembler und nur in seltenen Notfällen müssen Sie den Offset berechnen. Zum anderen gibt es noch eine Faustregel, mit der man sich das ganze vereinfa-

chen kann. Die soll durch folgendes Schema erläutert werden:

Byte	Inhalt	Offset
...		
1995		F9
1996		FA
1997		FB
1998		FC
1999		FD
2000	BNE	FE
2001	Offset	FF
2002	Programmzählerstand	
2003		01
2004		02
2005		03

Bei Vorwärtssprüngen ist ohnehin alles klar: Bei einem Sprung nach Adresse 2005 müßte man in vorliegendem Fall einen Offset von 03 eingeben. Bei Rückwärts-Verzweigungen zählt man einfach von $FF an rückwärts bis zur Zieladresse. Eine Verzweigung nach 1996 würde im vorliegenden Fall also einen Offset von $FA erfordern.

Eine Einschränkung der relativen Adressierung können Sie nun auch sofort verstehen, wenn Sie an Zweierkomplementzahlen denken: Der Offset belegt ein Byte. Die größte positive Zahl in einem Byte ist
0111 1111 = +127 = $7F
und die kleinste negative Zahl ist
1000 0000 = —128 = ($80)

Es sind keine größeren Vorwärts-Verzweigungen als um 127 Bytes möglich, weil in diesem Fall ein Offset größer als $7F, also mit einem Bit 7 gleich 1 nötig wäre, was aber wieder als negative Zweierkomplementzahl verstanden und einen Rückwärtssprung verursachen würde. Ähnliches gilt anders herum: Es ist kein weiterer Rücksprung als um 128 Bytes möglich, weil das im Offset zum gelöschten Bit 7 führen würde, also zu einem Offset kleiner als $80, was wiederum anstelle des Rücksprunges eine Vorwärts-Verzweigung herbeiführen würde.

Darauf sollte man achten beim Erstellen eines Assembler-Programmes, daß man nie weitere Rückwärtssprünge als um 128, beziehungsweise Vorwärtssprünge um 127 Bytes verlangt. Auch wenn man im Assembler gar nicht auf relative Adressierung Rücksicht nehmen muß, weil der Assembler sich mit den Absolutadressen begnügt, sollte man wissen, daß zum Beispiel folgende Zeile aufgrund dieser Einschränkung nicht möglich ist:
3000 BNE 1000

Die meisten Assembler reagieren auf solch eine Zeile mit einer Fehlermeldung oder so wie der SMON, der klammheimlich die Programmstartadresse statt 1000 einsetzt. Aber es ist doch ärgerlich, wenn man auf dem Papier ein Programm fertig hat und erst beim Eintippen feststellt, daß der Computer das so nicht haben will.

Zeropage-Adressierung

Weil wir nun gerade mit der Adressierung so schön in Schwung sind, stelle ich Ihnen noch eine andere vor: Die Adressierung der Zeropage. Was ist die Zeropage? Auf deutsch heißt das Nullseite. Am besten versteht man das, wenn man sich in Erinnerung ruft, wie Adressen in unserem Computer verwaltet werden. Da haben wir doch ein LSB (Least Significant Byte) und ein MSB (Most Significant Byte), zum Beispiel $1F 04 (mit 1F als MSB und 04 als LSB). Nun hat unser C 64 65535 Adressen von $0000 bis $FFFF. Bei den ersten 256 Adressen von $0000 bis $00FF ist das MSB $00. Man nennt so einen 256-Byte-Block eine Seite (engl. page). Weil hier für alle Adressen dieser ersten Seite des MSB Null ist heißt sie Nullseite = Zeropage. Messerscharf werden Sie schließen, daß man die Seite mit den MSBs $01 als erste Seite bezeichnet, die mit den MSBs $02 als 2. Seite und so weiter.

Wenn wir nun zum Beispiel den Akku mit dem Inhalt der Zeropage-Adresse $00FA laden wollen, dann könnten wir schreiben:
3000 LDA 00FA

Unser Mikroprozessor versteht uns aber auch, wenn wir nur schreiben:
3000 LDA FA

Das ist sie, die Zeropage-Adressierung. Anstelle eines 3-Byte-Befehls ist das jetzt ein 2-Byte-Befehl, was Speicherplatz und vor allem Rechenzeit einspart. Auf diese Weise kann man von den bisher kennengelernten Befehlen folgende adressieren:
LDA, LDX, LDY, STA, STX, STY, INC, DEC, ADC und SBC

Sie können sich merken, daß man (bis auf zwei Ausnahmen, die wir noch kennenlernen werden) alle absolut adressierbaren Befehle auch Zeropage-absolut anwenden kann. Genauere Angaben über die Codes, die Ausführungszeiten und die Beeinflussung der Flaggen (letztere ist identisch mit der absoluten Adressierung) entnehmen Sie bitte der angefügten Tabelle 1.

Zum Thema Geschwindigkeit: Wenn Sie die benötigten Taktzyklen von absolut und von 0-absolut adressierten Befehlen in den Tabellen miteinander vergleichen, werden Sie jeweils einen Unterschied von einem Zyklus feststellen. Das mag Ihnen läppisch vorkommen. Bedenken Sie aber, daß Sie sehr häufig Schleifen programmieren müssen, die mehrere 100 Mal durchlaufen werden, die vielleicht als oft zu verwendende Unterprogramme dienen... Sie werden bald feststellen, daß da schnell beachtliche Zeitunterschiede auftreten können: Für zeitkritische Programme ist die Verwendung der Zeropage-Adressierung dringend geboten.

Dieser Tatsache waren sich leider auch die Schöpfer unseres Betriebssystems und des Basic-Interpreters voll bewußt. Die Zeropage ist nahezu randvoll mit Speicherstellen, in denen sich beide Programmkomplexe tummeln. Fast jede Kernal- und Interpreter-Routine notiert sich irgendwelche Werte auf der Seite Null. Das macht es uns als Assembler-Programmierer nicht gerade leicht, die Zeropage-Adressierung zu verwenden, wenn wir außerdem den Interpreter oder das Betriebssystem benutzen wollen. Es kann geradezu katastrophale Folgen

Befehlswort	Adressierung	Byteanzahl	Code Hex	Code Dez	Dauer in Taktzyklen	Beeinflussung von Flaggen
LDA	0-Page, abs.	2	A5	165	3	N,Z
LDX	0-Page, abs.	2	A6	166	3	N,Z
LDY	0-Page, abs.	2	A4	164	3	N,Z
STA	0-Page, abs.	2	85	133	3	—
STX	0-Page, abs.	2	86	134	3	—
STY	0-Page, abs.	2	84	132	3	—
INC	0-Page, abs.	2	E6	230	5	N,Z
DEC	0-Page, abs.	2	C6	198	5	N,Z
ADC	0-Page, abs.	2	65	101	3	N,V,Z,C
SBC	0-Page, abs.	2	E5	229	3	N,V,Z,C
CMP	unmittelbar	2	C9	201	2	N,Z,C
	absolut	3	CD	205	4	N,Z,C
	0-Page, abs.	2	C5	197	3	N,Z,C
CPX	unmittelbar	2	E0	224	2	N,Z,C
	absolut	3	EC	236	4	N,Z,C
	0-Page, abs.	2	E4	228	3	N,Z,C
CPY	unmittelbar	2	C0	192	2	N,Z,C
	absolut	3	CC	204	4	N,Z,C
	0-Page, abs.	2	C4	196	3	N,Z,C

Tabelle 1: Kenndaten der neuen Befehle und Adressierungen

lsn $	msn	$ bin. binär	0	1	2	3	4	5	6	7
			0000	0001	0010	0011	0100	0101	0110	0111
0	0	000	NUL	DLE	SP	0	@	P		p
			NULL	DLE	SP	0	@	P	CHR$(96)	CHR$(112)
1	0	001	SOH	DC1	!	1	A	Q	a	q
			SOH	DC1	!	1	A	Q	CHR$(97)	CHR$(113)
2	0	010	STX	DC2	"	2	B	R	b	r
			STX	DC2	"	2	B	R	CHR$(98)	CHR$(114)
3	0	011	ETX	DC3	#	3	C	S	c	s
			ETX	DC3	#	3	C	S	CHR$(99)	CHR$(115)
4	0	100	EOT	DC4	$	4	D	T	d	t
			EOT	DC4	$	4	D	T	CHR$(100)	CHR$(116)
5	0	101	ENQ	NAK	%	5	E	U	e	u
			ENQ	NAK	%	5	E	U	CHR$(101)	CHR$(117)
6	0	110	ACK	SYN	&	6	F	V	f	v
			ACK	SYN	&	6	F	V	CHR$(102)	CHR$(118)
7	0	111	BEL	ETB	'	7	G	W	g	w
			BEL	ETB	'	7	G	W	CHR$(103)	CHR$(119)
8	1	000	BS	CAN	(	8	H	X	h	x
			BS	CAN	(	8	H	X	CHR$(104)	CHR$(120)
9	1	001	HT	EM	)	9	I	Y	i	y
			HT	EM	)	9	I	Y	CHR$(105)	CHR$(121)
A	1	010	LF	SUB	*	:	J	Z	j	z
			LF	SUB	*	:	J	Z	CHR$(106)	CHR$(122)
B	1	011	VT	ESC	+	;	K	[	k	{
			VT	ESC	+	;	K	[	CHR$(107)	CHR$(123)
C	1	100	FF	FS	,	<	L	\	l	¦
			FF	FS	,	<	L	£	CHR$(108)	CHR$(124)
D	1	101	CR	GS	—	=	M	]	m	}
			CR	GS	—	=	M	]	CHR$(109)	CHR$(125)
E	1	110	SO	RS	.	>	N	↑	n	~
			SO	RS	.	>	N	↑	CHR$(110)	CHR$(126)
F	1	111	SI	US	/	?	O	—	o	DEL
			SI	US	/	?	O	←	CHR$(111)	CHR$(127)

Bild 2: ASCII-Code (jeweils oben) und Commodore-ASCII-Code (jeweils unten) (msn = most significant nibble; lsn = least significant nibble)

haben, einige Zeropage-Adressen zu überschreiben. Andere werden ständig neu beschrieben durch das Betriebssystem oder den Interpreter, was unseren eigenen — vielleicht gerade in so einer Speicherzelle gelagerten — Zwischenwerten den Garaus machen würde. Man sollte sich also die ersten 256 Speicherstellen ganz genau ansehen, bevor man sie adressiert oder aber auf das Betriebssystem und den Basic-Interpreter verzichten. Ersteres erleichtern Tabellen der Speicherbelegung (zum Beispiel Babel, Krause, Dripke »Das Interface Age Systemhandbuch zum Commodore 64«, Interface Age Verlag, oder »Das Commodore 64 Buch, Band 4, Ein Leitfaden für Systemprogrammierer«, Markt und Technik Verlag) und auch die Serie von Dr. Helmut Hauck »Memory Map mit Wandervorschlägen«, die seit Ausgabe 11/84 erscheint.

Ohne Hemmungen nutzen dürfen wir nur die Speicherstellen (jedenfalls beim C 64) $02 und $FB bis $FE. Weil das doch recht mikrig ist, hat jeder Assembler-Programmierer spezielle Tips, welche Zellen er noch mit welchen Vorsichtsmaßnahmen benutzt. Wenn man bestimmte Routinen aus dem Betriebssystem oder dem Interpreter nicht aufruft, bleiben dazugehörige Zeropage-Adressen unbeeinflußt und sind dann für eigene Zwecke nutzbar. Manchmal ist es notwendig, den alten Zustand einer Adresse nach Beendigung eigener Programme wieder herzustellen, manchmal nicht. Interessant und viel beschrieben in allen möglichen Zeitschriften, Büchern etc. ist die Möglichkeit, die Notizen, die sich das Betriebssystem oder der Interpreter auf der Zeropage macht, zu verändern. Im Prinzip schreibt man damit kleine Teile dieser Großprogramme um oder variiert Tabellenteile davon. Wie schon Dr. Hauck in seiner Serie sagt, geschieht das im Rahmen der »Tricks« mit irgendwelchen POKEs mehr oder weniger blind, weshalb auch bevorzugt Abstürze des Computers dabei festzustellen sind.

FLAGGE	Akku X Y } > DATEN	Akku X Y } = DATEN	Akku X Y } < DATEN
N	0	0	1
Z	0	1	0
C	1	1	0

Bild 1. Flaggen bei den Vergleichsbefehlen

Warum Abstürze? Na, stellen Sie sich mal ein von Ihnen geschriebenes Programm vor — zum Beispiel das aus der letzten Ausgabe, zur Berechnung der Summe einer arithmetischen Reihe — und POKEn Sie dann anstelle irgendeines Befehlscodes, der dorthin gehört, jetzt eine 0 (also ein BRK) hinein. Die Wirkung dürfte ähnlich sein. Wenn man allerdings die Funktion der betreffenden Speicherstelle genau kennt, lassen sich recht nützliche Änderungen hervorrufen, wie zum Beispiel die SchutzPOKEs für den Basic-Speicher durch Verändern der Adressen $33, $34, $37 und $38.

Wir werden im folgenden immer dann, wenn wir mit Zeropage-Adressierung arbeiten oder Routinen des Betriebssystems oder Interpreters untersuchen, spezielle Stellen der Nullseite kennenlernen.

Vorhin hatte ich noch angedeutet, daß man dann die Zeropage fast vollständig nutzen könne, wenn man auf den Basic-Interpreter und das Betriebssystem verzichtet. Das ist tatsächlich möglich. Nur wird man dann erstaunt feststellen, wieviel Arbeit uns die computerinterne Software abnimmt oder anders herum: Viele bislang selbstverständliche Dinge werden wir dann plötzlich selbst programmieren müssen, und das kann ein hartes Brot sein!

Als Beispiel für ein Programm, das nicht nur die Zeropageadressierung verwendet, sondern sogar selbst komplett in der Zeropage steht, werden wir uns die CHRGET-Routine ansehen. Eine Klasse von Befehlen, die dort angewendet wird, die Vergleichsbefehle, soll zuvor noch gezeigt werden.

Die Vergleichsbefehle: CMP, CPX, CPY

Vergleichen heißt in englischer Sprache »to compare«, woraus Sie unschwer erkennen können, woher die Bezeichnung CMP und die CPs in CPX beziehungsweise CPY kommen. Verglichen wird jeweils der Akku-Inhalt (bei CMP), der Inhalt des X- (bei CPX) oder des Y-Registers (bei CPY) mit Daten, die der Compare-Befehl adressiert. Einige Beispiele werden Ihnen das klarer machen:

CMP #FF

vergleicht den Akku-Inhalt mit der Zahl $FF. Hier liegt die unmittelbare Adressierung vor, die ebenso für CPX und CPY verwendbar ist. Außerdem ist das dann ein 2-Byte-Befehl.

CPX 3000

vergleicht den Inhalt des X-Registers mit dem Inhalt der Speicherstelle $3000. Die absolute Adressierung ist also auch anwendbar (natürlich auch für CMP und CPY). Der Compare-Befehl besteht so aus 3 Bytes.

CPY A8

vergleicht den Inhalt des Y-Registers mit dem Inhalt der Zeropage-Stelle $A8. Diese soeben frisch gelernte Zeropage-Adressierung ist bei allen drei Vergleichsbefehlen möglich und macht aus ihnen 2-Byte-Befehle.

Für CPX und CPY sind das alle Möglichkeiten der Adressierung. CMP erlaubt weitere, die wir noch kennenlernen werden. Nun interessiert uns natürlich noch, wie das Vergleichsergebnis zu erhalten ist! Bei diesen Befehlen geschieht merkwürdiges: Die Vergleichsdaten werden vom Inhalt des Akkus (beziehungsweise X- oder Y-Registers) abgezogen, aber: Weder wird dieser Inhalt noch werden die adressierten Daten verändert! Der Trick ist, daß drei Flaggen das Ergebnis anzeigen: Die Negativ-Flagge N, die Null-Flage Z und das Carry-Bit C. Diese Anzeige geschieht so:

1) Der Registerinhalt (Akku, X-, Y-Register) ist größer als die Vergleichsdaten:
Dann ist das Carry-Bit = 1, die N- und die Z-Flagge = 0.

2) Der Registerinhalt ist gleich den Vergleichsdaten:
Dann sind Carry- und Z-Flagge = 1, die N-Flagge = 0.

3) Der Registerinhalt ist kleiner als die Vergleichsdaten:
Die N-Flagge ist dann = 1, Carry- und Zero-Flagge sind 0.

Damit Sie die Übersicht behalten können, ist in Bild 1 das ganze als Schema gezeigt.

Sie werden sich vermutlich schon denken können, wie der Hase weiterläuft: Mit den Verzweigungsbefehlen prüfen wir die Flaggen und springen die gewünschten weiteren Programm-Routinen an.

Die Kombination der Compare-Befehle mit den Verzweigungsoperationen wird Ihnen im weiteren Verlauf dieses Kurses noch ganz geläufig werden. Ein Beispiel sehen Sie nachher ebenfalls in der CHRGET-Routine. Leider muß ich Sie immer noch etwas vertrösten, denn mit Verstand begreifen läßt sich diese Routine nur dann, wenn man etwas mehr über die Codierung von Zeichen weiß. Deswegen werden wir uns nun noch mit dem ASCII-Code und dem Commodore-ASCII herumschlagen.

Zeichencodierung mit dem ASCII- und dem Commodore-ASCII-Code

ASCII ist die Abkürzung von »American Standard Code for Information Interchange« und das heißt auf deutsch »amerikanischer Standard-Code zum Informations-Austausch«. Diese Zeichenverschlüsselungsart ist international als ISO-7-Bit-Code genormt, und es wäre wirklich nett, wenn alle sich daran halten würden. Tatsächlich aber finden wir zum Beispiel bei unserem C 64 eine Abart des Normcodes, den Commodore-ASCII-Code. Über die damit erzwungenen Umrechnungen können alle diejenigen Dramen erzählen, die zum erstenmal einen (Nicht-Commodore-)Drucker an ihr Gerät anschließen oder aber blauäugig in den Online-Betrieb mit anderen Computern eintreten wollten.

Sehen wir uns zunächst einmal den ASCII-Code an. Es handelt sich um einen 7-Bit-Code, das heißt 128 Zeichen können in nur 7 Bits untergebracht werden (0000 0000 bis 01111111). Das achte Bit dient bei manchen Operationen mit Computer-Peripherie als Paritäts-Bit. Bei dieser Gelegenheit soll auch gleich erklärt werden, was Parität in diesem Zusammenhang bedeutet. Wer-

NUL	Null	
SOH	Start of heading	Beginn des Kopfes
STX	Start of text	Textbeginn
ETX	End of text	Textende
EOT	End of transmission	Übertragungsende
ENQ	Inquiry	Anfrage
ACK	Acknowledge	Bestätigung
BEL	Bell	Klingel
BS	Backspace	Zurücksetzen
HT	Horizontal tabul.	Horizontaltabulator
LF	Line feed	Zeilenvorschub
VT	Vertical tabulator	Vertikaltabulator
FF	Form feed	Formatvorschub
CR	Carriage return	Wagenrücklauf/Zeilenwechsel
SO	Shift out	Rückschaltung
SI	Shift in	Dauerumschaltung
DLE	Data link escape	Datenverbindungsumschaltung
DC1-4	Device control	Gerätesteuerung
NAK	Negative acknowl.	Negativ-Bestätigung
SYN	Synchronous idle	Synchronisations-Leerlauf
ETB	End of transmission block	Ende des Übertragungsblockes
CAN	Cancel	Annullieren
EM	End of medium	Datenträgerende
SUB	Substitute	Ersetzen
ESC	Escape	Umschaltung
FS	File separator	Dateitrennzeichen
GS	Group separator	Gruppentrennzeichen
RS	Record separator	Satztrennzeichen
US	Unit separator	Einheiten-Trennz.
SP	Space	Leerzeichen
DEL	Delete	Löschzeichen

Bild 4. Die Bedeutung der Abkürzungen im ASCII-Code

den Daten übertragen, muß immer mit Übermittlungsfehlern gerechnet werden. Das Paritätsbit dient dazu festzustellen, ob ein Byte korrekt angekommen ist. Bei der sogenannten geraden Parität zählt man die Einser im Byte zusammen und setzt Bit 7 auf 1 wenn sich eine ungerade Zahl ergibt. Mit dem Paritätsbit haben wir dann eine gerade Zahl. Ist die Quersumme des Byte schon gerade, bleibt Bit 7 eine Null. Ebensogut kann man die ungerade Parität verwenden, indem dann Bit 7 so gewählt wird, daß sich immer eine ungerade Zahl ergibt. Welche Art der Parität zur Anwendung kommt, ist Vereinbarungssache. Nehmen wir mal an, es sei gerade Parität gefordert und ein Byte mit der Information 00010110 soll übermittelt werden. Die Quersumme ist 3, also ungerade. Das Paritätsbit muß auf 1 gesetzt werden. Wir senden das Byte 10010110. Der Empfänger überprüft zunächst auf gerade Parität und verwendet dann nur die Bits 0 bis 6. Doppelfehler, die mittels des Paritätsverfahrens nicht festgestellt werden können, sind sehr selten. Leider kann auf diese Weise nur bemerkt werden, daß ein Übertragungsfehler aufgetreten sein muß, aber nicht welcher. Die Information muß dann neu angefordert werden.

Sehen wir uns nun den Commodore-ASCII-Code an. Durch die Einbindung der Grafikzeichen brauchen wir mehr als die 128 Kombinationen. Commodore benutzt deswegen einen 8-Bit-Code. Mit dem Basic-Befehl CHR$(x) können Sie sich alle 256 Möglichkeiten ansehen. Erschwerend kommt aber noch hinzu, daß wir nicht nur einen Zeichensatz, sondern deren vier zur Verfügung haben, die durch den jeweiligen Schreibmodus ansprechbar sind (Klein-/Großschriftmodus, Großschriftmodus, beide Modi mit Reverse-ON oder OFF). Im Zeichen-ROM liegen insgesamt 512 Muster abrufbereit. Zu diesen kommen beim CHR$-Befehl noch eine ganze Reihe von Steuerzeichen hinzu... die Verwirrung ist perfekt! Wir wollen an dieser Stelle keine Entwirrung vornehmen, sondern wir durchschlagen den Gordischen Knoten, indem wir nur die ersten 128 Zeichen mit den ASCII-Zeichen vergleichen. In Bild 2 und 3 finden Sie unsere Gegenüberstellung.

Einige Kombinationen dienen als Steuer-Codes. (Die Bedeutung der dabei verwendeten Abkürzungen sehen Sie unten.

Nur ein Teil dieser Codes wird tatsächlich genutzt. Andere haben — je nach Gerät an das sie gesandt werden — unterschiedliche Bedeutungen. Denken Sie dabei nur mal an die verschiedenen Betriebssysteme des Commodore-Druckers 1526, wo man bei dem einen mit CHR$(1), bei dem anderen mit CHR$(14) den Breitschrift-Modus anschaltet. Innerhalb unseres Computers werden offensichtlich bestimmte Codes anders genutzt. Das sind:

Anstelle von	geschieht folgendes:
ENQ	Zeichen weiß
BS	Blockieren der Umschaltung Klein-/Großschrift
HT	Zulassen der obigen Umschalt.
DC1	Cursor abwärts
DC2	Reverse-Modus an
DC3	Cursor in HOME-Position
DC4	INST/DEL
FS	Zeichen rot
GS	Cursor rechts
RS	Zeichen grün
US	Zeichen blau

Der auffälligste Unterschied ist der, daß beim Commodore-ASCII anstelle der Kleinbuchstaben Grafikzeichen liegen. Sollte anstelle des Normalmodus der Klein-/Großschriftmodus eingeschaltet sein, findet man anstelle der Großbuchstaben die kleinen.

Jetzt haben wir alle nötigen Kenntnisse, um die CHRGET-Routine in unserem Computer zu verstehen.

Die CHRGET-Routine

Das Kürzel CHRGET kommt von »Get a character«, was bei uns heißt: »Hole ein Zeichen«. Es handelt sich um eine sehr häufig benutzte Routine unseres Basic-Interpreters, die — wie schon vorhin erwähnt — komplett in der Zeropage steht. Wenn Sie mit dem SMON mal nachsehen wollen, dann geben Sie den Befehl

D 0073 008B

ein. Sie haben dann die komplette Routine vor sich:

```
0073  E6  7A    INC 7A
0075  D0  02    BNE 0079
0077  E6  7B    INC 7B
0079  AD  2502  LDA 0225
007C  C9  3A    CMP #3A
007E  B0  0A    BCS 008A
```

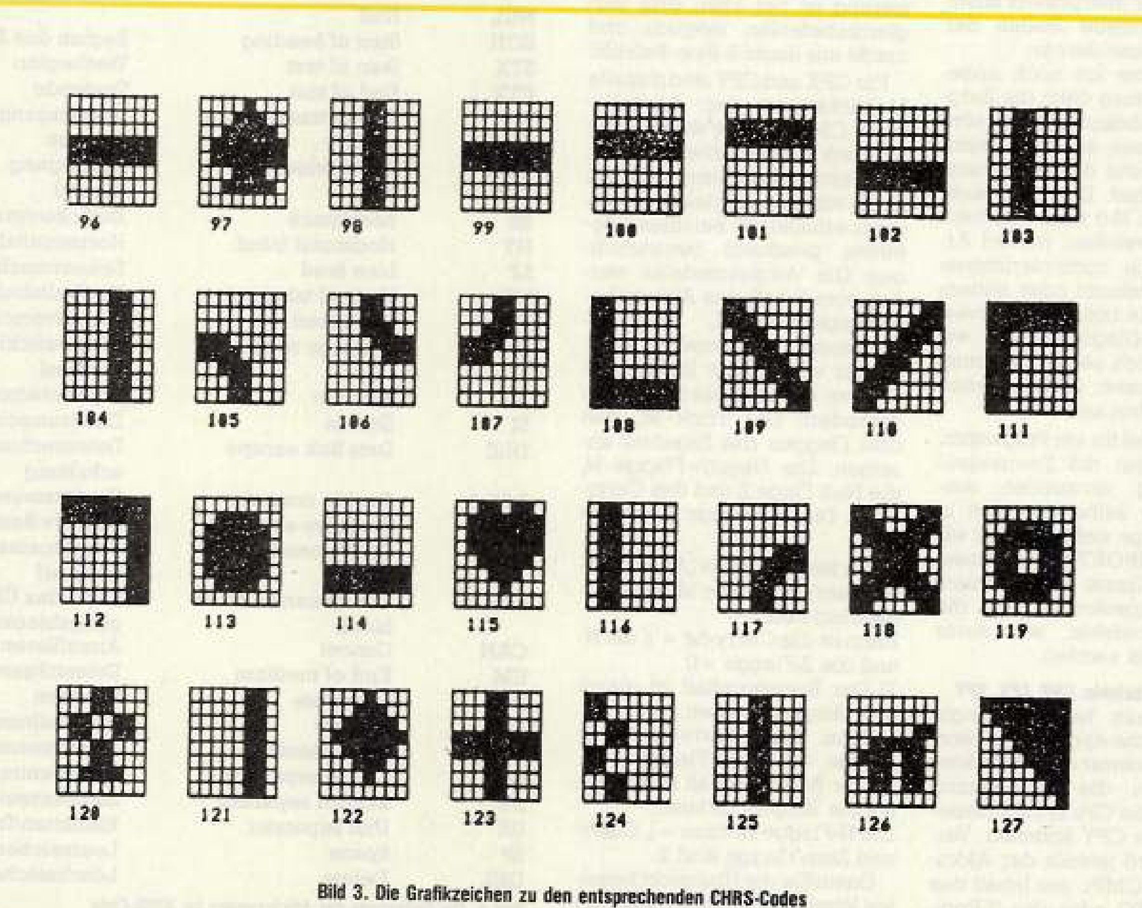


Bild 3. Die Grafikzeichen zu den entsprechenden CHR$-Codes

```
0080  C9  20   CMP #20
0082  F0  EF   BEQ 0073
0084  38  SEC
0085  E9  30   SBC #30
0087  38  SEC
0088  E9  D0   SBC #D0
008A  60  RTS
```

Eventuell sieht die Zeile 0079 bei Ihnen anders aus. Das liegt dann an den Speicherstellen 7A und 7B, welche einen Zeiger darstellen (LSB = 7A und MSB = 7B), der bei Ihnen gerade auf einen anderen Platz zeigt als auf $0225.

Diese CHRGET-Routine besteht aus drei Teilen:
Zeilen 0073 bis 0079
Weiterstellen des CHRGET-Zeigers und Einladen des dadurch angezeigten Speicherzelleninhaltes in den Akku.
Zeilen 007C bis 0082
Prüfroutinen
Zeilen 0084 bis 008A
Flaggen-Routinen

Im ersten Teil haben wir schon gleich etwas neues vor uns: Ein sich selbst veränderndes Programm. Die Speicherstelle (aus dem Basic-Eingabepuffer), aus der der Akku ein Zeichen holt, wird um 1 weitergezählt mit INC 7A.

Dabei handelt es sich um das LSB der Adresse und die nächste Zeile prüft, ob ein Überlauf (255+1) stattgefunden hat: BNE 0079.

Diese Technik kennen wir schon aus den letzten Folgen: Bei Überlauf wird die Z-Flagge auf 1 gesetzt und der BNE-Befehl führt keinen Sprung herbei. Den Offset von 02 können wir leicht nachrechnen: Der Programmzähler steht schon auf 0077. Die Zieladresse 0079 ist also noch 2 Bytes entfernt. Hat eine Überschreitung des Höchstwertes 255 stattgefunden, dann muß das dazugehörige MSB um 1 erhöht werden. Dies tut die nächste Zeile: INC 7B

In beiden Fällen ist nun der Zeiger 7A/7B um eine Stelle weitergerückt und der Inhalt der dadurch angezeigten Speicherstelle wird in den Akku geladen. Zwei Dinge können wir uns aus diesem kurzen Programmteil merken:

1) Wie man eine 16-Bit-Zahl hoch- (oder auch herunter-) zählt und
2) eine Möglichkeit, Zeiger einzusetzen. Wir werden noch eine Reihe anderer Zeigertypen kennenlernen und sehen, daß es nicht immer so direkt zugeht wie hier.

Im zweiten Teil finden wir die Prüfroutinen. Die Vergleichsbefehle beschränken sich auf den Akkuinhalt, also CMP.

CMP #3A testet, in welcher Beziehung das im Akku befindliche Zeichen zum Wert $3A = dezimal 58 steht. Erinnern wir uns an das Schema in Bild 1:

1) Commodore-ASCII-Code im Akku größer als 58, also Zeichen hinter dem Doppelpunkt (Buchstaben, Grafikzeichen, einige Sonderzeichen). Dann ist die Carry-Flagge = 1, N- und Z-Flagge sind 0.
2) Im Akku steht genau der Code 58, also der Doppelpunkt. Dann sind Carry-Bit und Z-Flagge = 1, nur die N-Flagge = 0.
3) Der Code des Zeichens im Akku ist kleiner als 58 (das wären alle Zahlen, einige Sonderzeichen und Steuerzeichen). In diesem Fall ist die N-Flagge = 1. Die beiden anderen Flaggen zeigen Null.

Der nun folgende Befehl BCS 008A überprüft die Carry-Flagge. Wenn sie gesetzt ist, wenn also der Code im Akku größer oder gleich dem eines Doppelpunktes (58) ist, springt der Programmzähler zum RTS. Der Code (und auch die Flaggen) wird unverändert zum aufrufenden Hauptprogramm weitergegeben. Zur Übung können Sie ja nochmal den Offset nachrechnen. Der Rest des Programms wird nur noch durchlaufen, wenn Codes kleiner als 58 im Akku stehen.

Die nächste Zeile CMP #20 dient zum Vergleich des Space-Codes $20 = dezimal 32 (Leertaste). Die Flaggen treten dann, wie schon oben beim ersten Vergleich gezeigt, je nach Akku-Inhalt auf. Durch die Verzweigung BEQ 0073 erfolgt ein Rücksprung zum Beginn der CHRGET-Routine dann, wenn die Z-Flagge gesetzt ist, also ein Space-Code im Akku liegt. Somit werden die Leerzeichen einfach übersprungen und das nächste Zeichen geholt. Alle anderen Zeichen, die bis hierher durchgehalten haben, werden nun im letzten Teil der CHRGET-Routine einer Prozedur unterworfen, die ich Flaggen-Routine genannt habe.

Durch zwei aufeinanderfolgende Subtraktionen, die insgesamt den Wert im Akku unverändert lassen (es wird 256 abgezogen), wird die Carry-Flagge beeinflußt. Verfolgen wir, was da passiert:
SEC dient als Vorbereitung für die folgende Subtraktion.
SBC #30 zieht vom Akku-Inhalt $30 = dezimal 48 ab. Wir wissen inzwischen, daß das der Addition des Zweierkomplementes entspricht. Dieses ist (rechnen Sie mal nach!) 1101 0000.

Nehmen wir mal an, wir hätten den Code der Zahl 4 (also dezimal 52 oder $34) im Akku stehen. Die Rechnung sieht dann so aus:

52	0011	0100
	1101	0000
+		
(1)	0000	0100

Das Ergebnis ist also 4, der Übertrag wird vernachlässigt.

Als anderes Beispiel sei nun der Code für das Ausrufungszeichen im Akku (dezimal 33 = $21 = binär 0010 0001). Die Rechnung ist dann:

33	0010	0001
+	1101	0000
	1111	0001

Das Ergebnis ist —15.

Alle Codes, die nicht für Zahlen stehen, haben nach dieser Subtraktion ein negatives Ergebnis im Akku hinterlassen und durch das »Borgen« das Carry-Bit gelöscht.

Nun machen wir weiter ab Zeile 0087:
SEC
SBC #D0

Wir ziehen $D0 = dezimal 208 ab. Das Zweierkomplement ist: ...Doch da kommen wir ins Stocken! Denn dieses Zweierkomplement ist nicht mehr mit 8-Bit-Zahlen darzustellen. Schon die Zahl 208 im Binärformat (1101 0000) würde als negative Zahl angesehen werden, weil Bit 7 gleich 1 ist. Wir machen es uns einfach und sagen, daß sich das Zweierkomplement wie bisher bilden läßt, aber dabei das Carry-Bit mit einbezogen wird. Unser Zweierkomplement ist dann also: 0011 0000 und das Carry-Bit ist gelöscht. Nun nehmen wir unser erstes Beispiel. Dort war nach der Subtraktion im Akku eine 4 verblieben:

	0000	0100
+	0011	0000
	0011	0100

Das ist wieder unser ursprünglicher Wert dezimal 52 = $34 = Code für die Zahl 4. Das Carry-Bit bleibt gelöscht.

Im zweiten Beispiel mit dem Ausrufungszeichen stand noch im Akku eine —15:

	1111	0001
+	0011	0000
(1)	0010	0001

Da haben wir wieder den Code für das Ausrufungszeichen ($21 = dezimal 33) im Akku und ein gesetztes Carry-Bit. Was kommt also bei der CHRGET-Routine heraus?

1) Alle Zeichen außer dem Space werden unverändert an das aufrufende Programm über den Akku weitergegeben. Space wird unterdrückt.
2) Bei allen Zeichen außer bei den Zahlen ist das Carry-Bit gesetzt.
3) Manche der aufrufenden Routinen überprüfen außer dem Zustand der Carry-Flagge auch den der Z- oder N-Flagge, die ja beim ersten CMP-Befehl ebenfalls gesetzt werden. So liefert die CHRGET-Routine noch weitere Informationen.

In der einschlägigen Literatur stoßen Sie auch auf eine Routine, die CHRGOT genannt wird. Es handelt sich dabei ebenfalls um die hier beschriebene CHRGET-Routine, nur erfolgt der Einsprung nicht bei $0073, sondern bei $0079.

Der Zeiger $007A/7B wird in diesem Fall nicht weitergestellt. Das vorher schon einmal in den Akku geladene Zeichen wird damit noch einmal angesprochen (got ist die Vergangenheitsform von get).

Mit dem CHRGET-Programm haben wir eines der wichtigsten Unterprogramme unserer computerinternen Software kennengelernt. Will man sich Interpreter-Routinen zunutze machen, stolpert man ständig darüber. Außerdem aber liegt die CHRGET-Routine im RAM. Das bedeutet, daß wir sie ohne weiteres für unsere Zwecke verändern können.

Ein Beispiel für so eine Änderung hat Christoph Sauer in seiner Serie über den »gläsernen VC 20« in der Ausgabe 9 (Seite 158) gezeigt. Dort wird die CHRGET-Routine nach dem LDA angezapft und auf das Pi-Zeichen geprüft, das neuen Befehlen vorangestellt wurde. Sehen Sie sich das Programm dort (auf Seite 160f.) mal genau an, viel kann man durch Nachvollziehen fremder Programme für die eigene Programmiertechnik lernen. Wir werden im Verlauf dieser Serie noch andere Möglichkeiten behandeln, die CHRGET-Routine zu verändern.

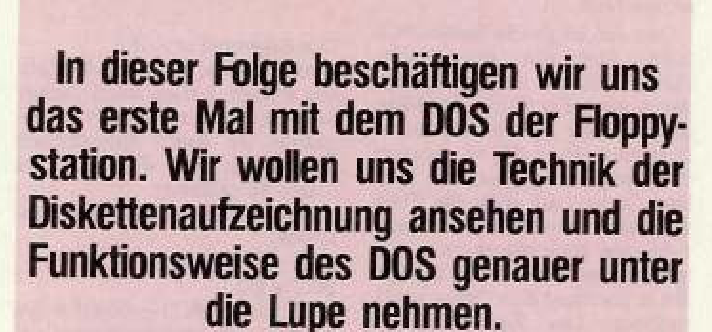
Damit sei es für diesmal genug. Als Assembler-Alchimisten gehören Sie jetzt zu den fortgeschrittenen Eleven, denn Sie können immerhin schon so trickreiche Programme wie die CHRGET-Routine nachvollziehen.

(Heimo Ponnath/gk)

In die Geheimnisse der Floppy



eingetaucht (Teil 4)



In dieser Folge beschäftigen wir uns das erste Mal mit dem DOS der Floppystation. Wir wollen uns die Technik der Diskettenaufzeichnung ansehen und die Funktionsweise des DOS genauer unter die Lupe nehmen.

Zuerst wollen wir uns mit dem Aufzeichnungsformat der Diskette beschäftigen: Für einen einwandfreien Betrieb der Floppystation ist es unumgänglich, daß sich Markierungen auf der Diskette befinden. Diese Markierung braucht das Laufwerk, um bestimmte Daten schnell finden zu können. Hierfür gibt es prinzipiell zwei Möglichkeiten: die Hardsektorierung und die Softsektorierung.

Hardsektorierte Disketten erkennt man daran, daß diese eine ganze Anzahl von Indexlöchern besitzen. Damit sind die kleinen Löcher nahe am Innenrand der Magnetscheibe gemeint. Mit einer Fotozelle können nun diese Löcher abgetastet werden, um die jeweilige Position der Diskette festzustellen. Dieses Verfahren hat den Vorteil, daß die Diskettenkapazität voll ausgenutzt werden kann. Es können so bis zu 5 MBytes Daten auf eine 5¼-Zoll-Diskette geschrieben werden. Allerdings erfordert diese Methode einen enormen Hardwareaufwand, der den Preis in die Höhe schnellen läßt. Für preiswerte Laufwerke (wie die 1541) geht man daher einen anderen Weg: die Softsektorierung. Hier besitzt die Diskette nur ein Indexloch zur Drehzahlüberwachung. Bei der 1541 ist sogar noch nicht einmal dieses erforderlich. Die notwendigen Markierungen werden beim Formatierungsvorgang softwaremäßig auf die Diskette aufgebracht, wobei natürlich wertvoller Speicherplatz verloren geht. Softsektorierte Disketten im 5¼-Zoll-Format verfügen daher über zur Zeit maximal 1 MByte Speicherkapazität.

Uns soll also im weiteren die Softsektorierung beschäftigen, wobei in Bild 1 eine Diskette schematisch dagestellt ist, nachdem sie auf der 1541 formatiert wurde. Sie ist in 35 konzentrische Spuren, nachfolgend Tracks genannt, aufgeteilt. Jeder dieser Tracks enthält wiederum eine bestimmte Anzahl von Sektoren, die von außen nach innen abnimmt. Diese Tatsachen sind Ihnen aber schon aus der ersten Folge bekannt. Nun wollen wir genauer auf den Aufbau der Sektoren einer Diskette eingehen.

Jeder Sektor besteht aus einem Blockheader und dem dazugehörigen Datenblock; eine schematische Darstellung zeigt Bild 2. Angeführt werden die Sektoren einer Diskette von den schon erwähnten Markierungen, die der Orientierung dienen. Diese Marken bezeichnet man als Synchron (SYNC)-Markierungen, sie bestehen aus mehreren $FF auf der Diskette. Erkennt der Schreib-/Lesekopf der Floppy also eine solche Marke, dann »weiß« die Foppystation, daß entweder ein Blockheader oder ein Datenblock nachfolgt. Nun müssen wir nur noch diese beiden voneinander unterscheiden können.

Hierzu dient das nächste Kennzeichen auf Diskette. Es folgt direkt nach der SYNC-Markierung und meldet dem Diskcontroller (DC) ob ein Blockheader oder ein Datenblock vorliegt. Hat das Kennzeichen den Wert $08, so handelt es sich um einen Blockheader; findet der Kopf hingegen den Wert $07, so handelt es sich um den Beginn eines Datenblocks.

Wir nehmen jetzt einmal an, der DC hätte das Kennzeichen $08 entdeckt; es handelt sich also um den Header eines Datenblocks. Dann folgt als nächstes Byte die Prüfsumme über den Header, die zur Kontrolle auf Lesefehler dient. Die Reihenfolge der Headerbytes, wie sie im Commodore-Handbuch angegeben ist, stimmt nicht mit der Aufzeichnung auf Diskette überein.

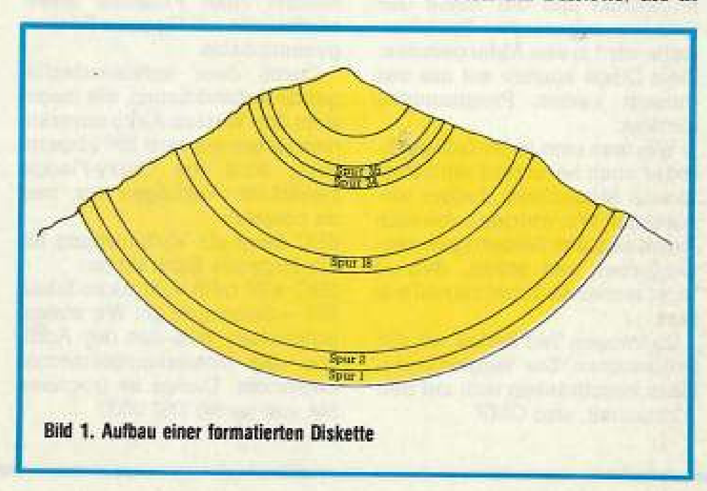


Bild 1. Aufbau einer formatierten Diskette

Die nächsten 2 Bytes stellen Sektor- und Tracknummer dieses Sektors dar. Anhand dieser Werte kann der DC bei Trackwechsel sehr schnell die Position des Schreib-/Lesekopfes ausfindig machen.

Das 5. und 6. Byte des Blockheaders geben jeweils einen Teil der ID der Diskette an, und zwar folgen zuerst das zweite und dann das erste Zeichen der ID, die beim Formatieren festgelegt wurden. Mit diesen Angaben ist die Behandlung des Headers bereits abgeschlossen. Es folgen jetzt noch ein paar Bytes, die eine Lücke darstellen.

Mit der nächsten SYNC-Markierung wird der Beginn des eigentlichen Datenblocks eingeleitet. Nach der SYNC-Marke folgt das Datenblockkennzeichen $07. Die nächsten zwei Bytes sind uns bestens bekannt. Sie können mit jedem Diskmonitor angesehen werden und geben Track- und Sektornummer des nächsten Blocks im File an. Man bezeichnet sie deshalb als Linker oder Linkadressen (engl.: to link = verbinden).

Nun erst folgen die eigentlichen Daten auf Diskette, die in

jedem Block 254 Byte ausmachen.

Hinter diesen Datenbytes steht die Prüfsumme des Datenblocks, die wiederum zum Erkennen von eventuellen Lesefehlern dient. Werden solche Fehler festgestellt, so versucht die Floppystation noch mehrere Male, den Block doch zu lesen. Erst wenn viele Versuche kein befriedigendes Ergebnis bringen, steigt sie mit einer Fehlermeldung aus.

Nach der Prüfsumme des Datenblocks folgt wieder eine Lücke auf der Diskette, bevor die SYNC-Markierung des nächsten Blockheaders kommt. Wenn wir uns diesen Aufbau eines Sektors betrachten, wird klar, warum die Speicherkapazität bei softsektorierten Disketten gegenüber hardsektorierten Disketten deutlich abnimmt.

Jetzt werden sie vielleicht auch die Beschreibung der Fehlermeldungen im Floppyhandbuch verstehen, die wir hier nicht mehr aufführen, da sie dort sehr genau und richtig erläutert werden.

Das Verständnis des Diskettenaufbaus ist für die weitere Behandlung des Floppy-DOS unerläßlich, da wir nur so die Funktionsweise begreifen lernen.

Jetzt wollen wir uns aber einmal mit der grundlegenden Arbeitsweise des Floppybetriebssystems (DOS) befassen, die um einiges komplizierter ist, als die im Computer.

Wenn wir die Floppy einschalten, passiert zunächst das gleiche, wie im Computer. Die RESET-Leitung geht auf Low und der Mikroprozessor, hier ein 6502, holt sich seine Systemstartadresse. Danach läuft das RESET-Programm an, wobei die Floppy einen Selbsttest durchführt. Erkennen können Sie dies daran, daß für kurze Zeit der Motor anläuft und die rote LED leuchtet. Wurde kein Defekt registriert, so erlischt die Leuchtdiode wieder, und der Motor geht aus. Jetzt wird der RAM-Bereich der Floppy initialisiert und alle wichtigen Zeiger werden hergestellt. Danach ist die 1541 betriebsbereit.

Von jetzt an laufen quasi drei Programme gleichzeitig ab:
- das Hauptprogramm läuft in einer Schleife, die nur bei der Ausführung von Befehlen verlassen wird;
- das Diskcontrollerprogramm wird über den IRQ gesteuert und durch den Timer des DC alle 10 ms aufgerufen;
- die Routinen des Buscontrollers (BC) schließlich, werden nur im Bedarfsfall aufgerufen, nämlich, wenn die ATN-Leitung des seriellen Bus auf Low geht.

Wir wollen uns die Funktion dieser Routinen nun einmal etwas genauer betrachten.

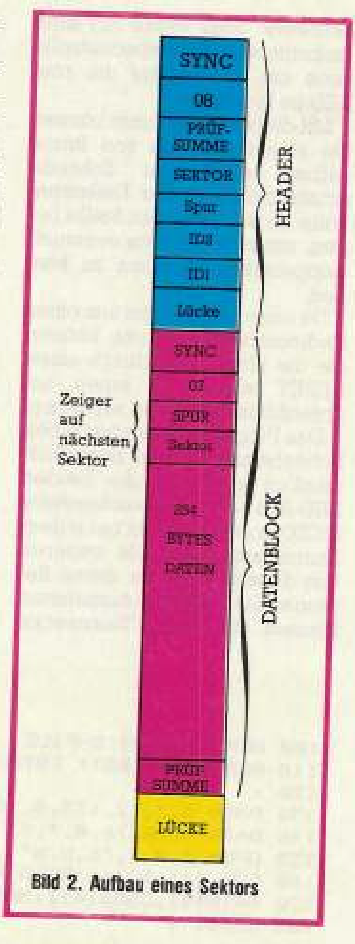


Bild 2. Aufbau eines Sektors

Das Hauptprogramm

Das Hauptprogramm hängt, wie schon gesagt, in einer Warteschleife, bis ein Befehl vom Computer kommt. Dieser aktiviert zuerst die Busroutinen, die die gesendeten Bytes dann entgegennehmen und abspeichern. Jetzt bekommt das Hauptprogramm, das übrigens den Zustand der beiden IRQ-Routinen (DC und BC) ständig überwacht, die Meldung, daß ein Befehl anliegt. Es verzweigt nun zur Befehlsauswertung, ähnlich dem Basic-Interpreter, und führt gegebenenfalls einen Befehl aus, sofern ein Syntaxfehler entdeckt wurde. In diesem Fall würde sonst eine Fehlermeldung generiert, die dann vom Computer ausgelesen werden kann.

Ist ein Befehl korrekt ausgeführt worden, so werden die Befehlsparameter wieder gelöscht, und das Hauptprogramm kehrt in die Warteschleife zurück.

Das Diskcontrollerprogramm

Der Diskcontroller enthält den Baustein VIA 6522, durch den er mit dem Mikroprozessor in Kontakt steht. Dieser Baustein enthält auch Timer, die in einem eingestellten Rhythmus einen IRQ auslösen können. Einer dieser Timer ist in der 1541 so eingestellt, daß er ungefähr alle 10 ms einen IRQ auslöst, der dann seinerseits das Diskcontrollerprogramm aufruft.

Es soll an dieser Stelle der Unterschied zwischen Diskcontroller und Diskcontrollerprogramm erläutert werden: Als Diskcontroller (DC) bezeichnet man die Hardware in der Floppy, die für den Laufwerksbetrieb zuständig ist.

Unter dem Diskcontrollerprogramm versteht man den Programmteil im DOS, der, durch IRQ geregelt, die Ansteuerung des DC übernimmt.

Eine vollständige Trennung dieser beiden Begriffe ist jedoch weder notwendig noch zweckmäßig, so daß wir mit dem Ausdruck »DC« immer die Gesamtheit von Hard- und Software beschreiben wollen. Nun aber wieder zu den Aufgaben des DC.

Auch dieses Programm hat eine Art Wartezustand, solange kein Befehl vom Computer anliegt. Wird nämlich das Hauptprogramm über den Bus aktiviert, so wertet dieses die Befehle aus und gibt sie an den DC weiter, der dann seinerseits dafür sorgt, daß das Laufwerk aktiviert wird. Er steuert den Laufwerk- und den Stepper (Schreib-/Lesekopf)-Motor und bedient die Daten, die vom und zum Tonkopf gehen. Die gesamten Vorgänge am Laufwerk werden also interruptgesteuert vorgenommen.

Die Busroutinen

Die Routinen des Buscontrollers (BC) werden ebenfalls über die IRQ-Leitung gesteuert. Auch der BC enthält einen VIA 6522-Baustein. Hier wird der Aufruf der Routinen allerdings nicht über den Timer organisiert, sondern, wie schon erwähnt, über die ATN-Leitung des seriellen Busses. Zieht der Computer also diese Leitung auf Low, so wird in der Floppy (und in allen anderen Peripheriegeräten ebenso) ein IRQ ausgelöst. Dann erfolgt die Abfrage, ob dieser IRQ vom Timer des DC kam. Ist dies nicht der Fall, so wird die BC-Routine aufgerufen, die dann den weiteren Busbetrieb übernimmt. Sollte die Floppy gerade einen Befehl bearbeiten, während schon ein neuer vom Computer gesendet wird, so wartet der BC so lang mit der Annahme, bis die Floppy wieder in den Bereitschaftszustand zurückgekehrt ist.

Wie Sie sehen, stellt das DOS eine ziemlich komplizierte Einheit dar, deren Schema in Bild 3 zu sehen ist.

> Wie Sie vielleicht bemerkt haben, ist uns in Folge 2 ein Fehler unterlaufen. Das abgedruckte Listing 6 wäre eigentlich Listing 5 gewesen. Als tatsächliches Listing 6 liefern wir Ihnen heute das Directory-Sortierprogramm nach, das Sie in Listing 1 abgedruckt finden.

```
100 REM DIRECTORY-SORTER
101 REM SORTIERT DIRECTORY ALPHABETISCH     <180>
102 REM BEI VIELEN EINTRAEGEN BITTE         <141>
103 REM ETWAS GEDULD (MAX. 5.MIN)           <226>
104 REM SORTIERT AUCH GESCRATCHTE FILES     <219>
105 REM MIT, STELLT SIE ABER NICHT          <052>
106 REM WIEDER HER ! SORTIERALGORITHMUS     <087>
107 REM KANN SICH IN EINEM SOLCHEN FALL     <048>
108 REM IN EINER ENDLOSSCHLEIFE VER-        <121>
109 REM HEDDERN. ABHILFE: NACH 3-4 MIN.     <039>
110 REM STOP-TASTE BRUECKEN, DANN           <009>
111 REM GOTO 210 EINGEBEN. SIND EINTR.      <143>
112 REM DANN NOCH NICHT VOLLKOMMEN SOR-     <019>
113 REM TIERT, NOCHMALS FUER EINIGE         <227>
114 REM MINUTEN LAUFEN LASSEN.              <236>
115 REM ACHTUNG !!! NUR ZUSAMMEN MIT        <208>
116 REM DEN UNTERPROGRAMMEN 1 & 2           <190>
117 REM ABLAUFFAEHIG !!!                    <233>
118 :                                       <182>
119 :                                       <176>
120 DIM DD$(144)                            <177>
130 MM=MM+1:GOSUB 1000                      <148>
140 IF DD$=NN$THEN MM=MM-1:GOTO 160         <203>
150 DD$(MM)=DD$:DD$="":GOTO 130             <248>
160 FOR GG=1 TO MM-1                        <190>
170 IF MID$(DD$(GG),4,16)<MID$(DD$(GG+1),4, <172>
   16)THEN 190
                                            <054>
180 HH$=DD$(GG):DD$(GG)=DD$(GG+1):DD$(GG+1)=HH$
   :FF=1
190 NEXT GG                                 <049>
200 IF FF THEN FF=0:GOTO 160                <206>
210 II=MM                                   <078>
220 FOR MM=1 TO II:DD$=DD$(MM):GOSUB 2000   <176>
   :NEXT MM
230 END                                     <030>
                                            <102>
```

Listing 1. Dieses Listing fehlte in Ausgabe 11/84

Wollen wir also in dieses System einsteigen, um dort eigene Programme ausführen zu lassen, so ist es natürlich unerläßlich, daß wir die »Spielregeln« dieses Prozessorsystems genau kennen, da es sonst leicht zu kleinen Katastrophen kommen kann.

Zu Ihrer weiteren Arbeit mit der 1541 noch ein paar Tips: Wenn Sie vorhaben, Programme in der Floppy ablaufen zu lassen, sollten Sie Ihre Floppy öffnen und ohne Deckel betreiben. So können Sie genau beobachten, wie der Kopf positioniert wird und was bei Lesefehlern geschieht. Sie werden unter anderem auch entdecken, daß Disketten nicht etwa auf der Seite beschrieben werden, auf der sich das Etikett befindet, sondern auf der Rückseite. Dies ist um so bemerkenswerter, als man eine Diskette immer nur auf der Vorderseite schonend behandelt, die ja eigentlich nicht benutzt wird. Auch wir mußten die Erfahrung machen, daß wir Disketten lange Zeit mit der wertvollen Seite auf Tische gelegt haben, stets darauf achtend, daß ja kein Staubkorn auf die von uns so gehütete Vorderseite kam.

Disketten werden auf ihrer Rückseite beschrieben!

Das Betreiben des Laufwerks ohne Deckel hat auch den Vorteil besserer Wärmeableitung. Die ICs werden es Ihnen danken.

Nachdem Sie Ihre 1541 also auf »Arbeitsbetrieb« getrimmt haben, wollen wir gleich einmal mit kleinen Programmen beginnen. In Tabelle 1 sehen Sie eine Aufstellung einiger wichtiger Zeropageadressen, die uns im weiteren Verlauf noch beschäftigen werden. Für ein DOS-Listing ist in unserer Serie natürlich kein Platz vorhanden; auch können wir nur mit kleinen Beispielen versuchen, Ihnen die Programmierung der Floppy nahezubringen. Für diejenigen unter Ihnen, die jedoch vorhaben, tiefer in die Floppyprogrammierung einzusteigen, sei an dieser Stelle ein Buch angesprochen, das voraussichtlich im Februar 1985 von Markt & Technik herausgegeben wird. Es behandelt die 1541 bis ins kleinste Detail, ist unter anderem mit einem ausführlichst kommentierten DOS-Listing ausgestattet und geht weit über das in dieser Reihe besprochene hinaus.

Programmieren der Floppy

So, jetzt soll es aber endlich losgehen. Wir wollen unser erstes Programm schreiben und in der Floppy ablaufen lassen.

Es handelt sich um Listing 2. Dieses »Miniprogramm« schreiben wir in den Puffer 0 der Floppy, das heißt ab Adresse $0300. Das Basic-Programm haben wir der Kürze halber gleich an den Assemblercode angehängt. Wenn Sie das Programm starten, wird das Bit abgefragt, das beim DC für den Zustand der Schreibschutzplakette verantwortlich ist. Sie werden vielleicht wissen, daß die Floppy die Schreibschutzkerbe bei den Disketten mit Hilfe einer Lichtschranke abfragt. Ist die Lichtschranke unterbrochen, das heißt es liegt eine Diskette mit Schreibschutzaufkleber im Laufwerk, dann steht das entsprechende Bit auf 0.

Unser Programm schiebt nun einfach das Bit der Lichtschranke an die Stelle des Bits für die rote LED und speichert diesen Wert wieder ab. Starten Sie einmal unser kleines Programm, dann werden Sie feststellen, daß die Leuchtdiode am Laufwerk erlischt, wenn die Lichtschranke unterbrochen wird. Holen Sie die Diskette dagegen aus dem Laufwerk oder legen Sie eine Diskette ohne Schreibschutzplakette ein, so beginnt die rote LED zu leuchten.

```
0 GOTO 10                                        <234>
1 , 0300 AD 00 1C LDA $1C00                      <018>
2 , 0303 29 10    AND #$10                       <023>
3 , 0305 4A       LSR                            <093>
4 , 0306 8D 00 1C STA $1C00                      <041>
5 , 0309 4C 00 03 JMP $0300                      <005>
6 :                                              <064>
10 OPEN 1,8,15                                   <208>
20 FOR X=0 TO 11:READ A                          <215>
30 PRINT#1,"M-W"CHR$(X)CHR$(3)CHR$(1)CHR$(A)
   :NEXT                                         <065>
40 PRINT#1,"M-E"CHR$(0)CHR$(3)                   <179>
50 DATA 173,0,28,41,16,74,141,0,28,76,0,3        <005>
```

Listing 2. Unser erstes Floppy-Maschinenprogramm

Mit diesem Programm können Sie also testen, ob von Ihnen selbst angefertigte Schreibschutzkerben in der Diskettenhülle an der richtigen Stelle liegen, um eine Diskette eventuell doppelseitig benutzen zu können.

Da unser Programm aus einer Endlosschleife besteht, können Sie die Floppy nur durch einen RESET wieder in einen ansprechbaren Zustand versetzen.

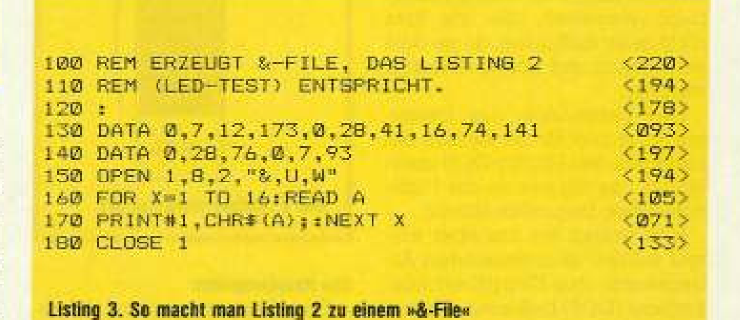
Das Programm hat aber einen Schönheitsfehler; es beeinflußt nämlich nicht nur die beiden LED-Bits in Speicherstelle $1C00, sondern löscht bei jedem Durchgang auch alle anderen Bits dieses Registers, deren Belegung Sie Tabelle 2 entnehmen können. Für unsere Testzwecke ist diese »Pfuscherei« jedoch unwesentlich.

Der »&«-Befehl

Nach diesem aufregenden Beispiel wollen wir Sie nun mit einem Befehl bekanntmachen, den Sie sehr wahrscheinlich noch nicht kennen. Er nennt sich »&« und wird unverständlicherweise in noch keinem uns bekannten Buch beschrieben. Der &-Befehl entspricht in gewisser Weise einem BLOCK-EXECUTE-Befehl; auch hier wird ein Programm von Diskette geladen und sofort ausgeführt.

Der Unterschied besteht nur darin, daß mit dem &-Befehl nicht nur ein Block, sondern ein ganzes File, das im Directory verzeichnet ist, geladen und im Puffer als Programm ausgeführt wird.

Außerdem müssen die Files, die mit dem Befehl »&« gestartet werden sollen, speziell gekenn-



```
100 REM ERZEUGT &-FILE, DAS LISTING 2            <220>
110 REM (LED-TEST) ENTSPRICHT.                   <194>
120 :                                            <178>
130 DATA 0,7,12,173,0,28,41,16,74,141            <093>
140 DATA 0,28,76,0,7,93                          <197>
150 OPEN 1,8,2,"&,U,W"                           <194>
160 FOR X=1 TO 16:READ A                         <105>
170 PRINT#1,CHR$(A);:NEXT X                      <071>
180 CLOSE 1                                      <133>
```

Listing 3. So macht man Listing 2 zu einem »&-File«

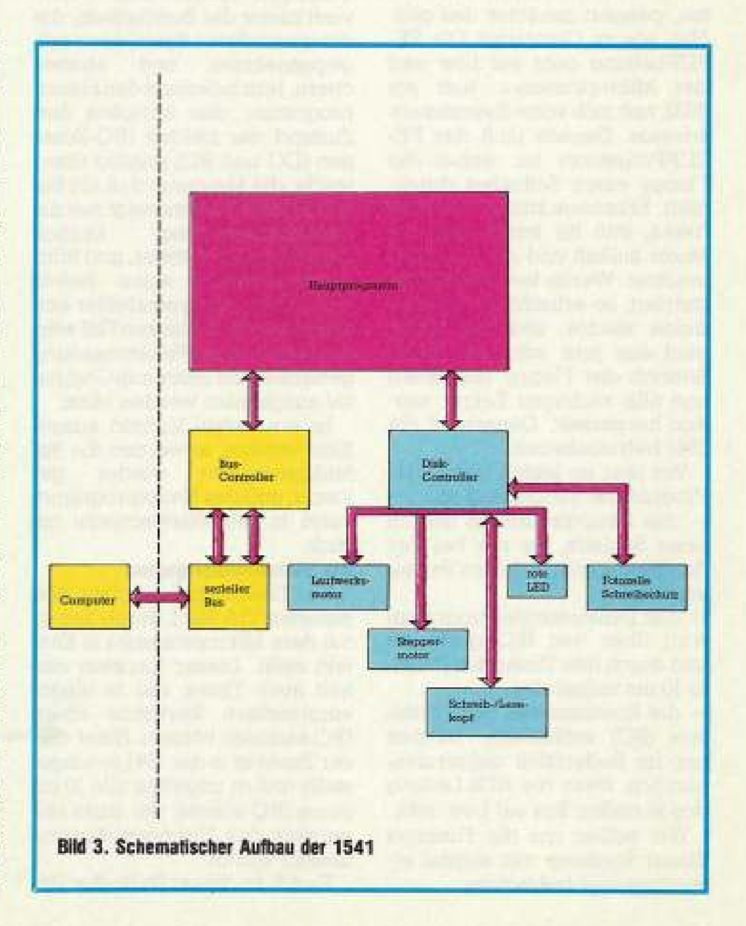


Bild 3. Schematischer Aufbau der 1541

Tabelle 1. Die wichtigsten Zeropageadressen der Floppy

```
#0000     Jobspeicher für Puffer 0
#0001     Jobspeicher für Puffer 1
#0002     Jobspeicher für Puffer 2
#0003     Jobspeicher für Puffer 3
#0004     Jobspeicher für Puffer 4
#0005     Jobspeicher für Puffer 5 (im RAM nicht vorhanden)
#0006/7   Spur- und Sektornummer für Befehl in Puffer 0
#0008/9   Spur- und Sektornummer für Befehl in Puffer 1
#000A/B   Spur- und Sektornummer für Befehl in Puffer 2
#000C/D   Spur- und Sektornummer für Befehl in Puffer 3
#000E/F   Spur- und Sektornummer für Befehl in Puffer 4
#0010/1   Spur- und Sektornummer für Befehl in Puffer 5
#0012/3   ID der Diskette im ASCII-Code: die beiden Zeichen der
          aktuellen ID werden bei jedem Blocksuchbefehl gelesen
          und hier aktualisiert abgespeichert. Auch das Initia-
          lisierkommando benutzt diesen Befehl und bringt die ID
          dadurch auf den neuesten Stand.
#0016-    Hier sind die Bytes für den aktuellen Blockheader ge-
#001A     speichert, und zwar sind dies:
           #0016  erstes Zeichen der ID
           #0017  zweites Zeichen der ID
           #0018  Spurnummer des Blocks
           #0019  Sektornummer des Blocks
           #001A  Prüfsumme über den Blockheader
          Auf der Diskette stehen diese Werte in der umgekehrten
          Reihenfolge!
#001C     Flag für Änderung beim Schreibschutz der Diskette
#002E/F   Zwischenspeicher für aktuelle Zeiger
#0030/1   Zeiger in aktuellen Puffer
#0032/3   Zeiger auf aktuellen Blockheader beim Schreiben
#0038     Kennzeichen (#07) für Beginn eines Datenblocks
#0039     Kennzeichen (#08) für Beginn eines Blockheaders
#003A     Zwischenspeicher für Prüfsummen
#003D     aktuelle Laufwerksnummer: bei der VC 1541 immer 0
#003E     gerade arbeitendes Laufwerk (#FF = kein Laufwerk)
#003F     Puffernummer des eben ausgeführten Befehls (0-5)
#0043     zählt die Anzahl der Sektoren bei der Formatierung
#0044     Zwischenspeicher beim Arbeiten
#0045     Zwischenspeicher für aktuellen Befehlscode
#0047     enthält aktuelles Kennzeichen für Beginn eines Daten-
          blocks. wird nur bei RESET einmal auf #07 gesetzt und
          kann vom Benutzer verändert werden, wobei das Hi-Nybble
          des Wertes immer auf 0 (#0-) stehen sollte, um Lesepro-
          bleme des DC zu vermeiden. Wird versucht, einen Daten-
          block mit einer anderen, als der hier gespeicherten,
          Nummer zu lesen, so erfolgt der Fehlercode #04 des DC
          und die Floppy sendet Fehlermeldung Nummer 22 zum Bus.
#0049     Zwischenspeicher für den Stackpointer
#004A     Zähler für Kopftransport; Zahlen bis 127 bewegen den
          Kopf nach außen; Zahlen von 128 bis 255 bewegen ihn
          nach innen (höhere Spurnummer).
#0051     aktuelle Spurnummer bei der Formatierung; steht auf
          #FF, wenn keine Formatierung erfolgt.
#0065/6   Zeiger auf die NMI-Routine; wird bei einem RESET ge-
          stellt.
#0067     Flag zum Anzeigen eines NMI
#0068     Flag zum Ermöglichen (0) oder Sperren (1) der automa-
          tischen Initialisierung einer Diskette, falls ein
          ID Type Mismatch Error erkannt wurde
#0069     Abstand der Sektoren bei der Zuteilung; erhält bei
          einem RESET den Wert 10.
#006A     Anzahl der Leseversuche eines Sektors; steht nach
          RESET auf 5.
#006B/C   Zeiger auf Sprungtabelle der USER-Befehle; steht nor-
          malerweise auf #FFF6 nach einem RESET.
#006D/E   Zeiger auf den Beginn der 'Bit Map'; steht auf #0400
          und wird beim Initialisieren gesetzt.
#006F     Zwischenspeicher; steht nach RESET auf #6F
#0070     Zwischenspeicher
#0071     Zwischenspeicher
#0072     Zwischenspeicher; steht nach RESET auf #FF
#0073     Zwischenspeicher
#0074     Zwischenspeicher
#0075/6   Indirekter Zeiger auf #0100; wird bei RESET gestellt
#0077     Gerätenummer + #20 für das LISTEN-Kommando
#0078     Gerätenummer + #40 für das TALK-Kommando
#0079     Flag für LISTEN (1/0)
#007A     Flag für TALK (1/0)
#007B     Flag für Adressierung
#007C     Flag für ATN-Signal vom seriellen Bus
#007D     Flag für Prozessor im ATN-Modus
#007F     Aktuelle Laufwerksnummer; hier immer 0
#0080     Aktuelle Spurnummer; enthält #00 nach Ausführung
#0081     Aktuelle Sektornummer; enthält #00 nach Ausführung
#0082     Aktuelle Kanalnummer
#0083     Aktuelle Sekundäradresse
#0084     übliche Sekundäradresse
#0085     Aktuelles Datenbyte
#0086     Speicher für Zwischenergebnisse
#0087     Speicher für Zwischenergebnisse
#0088     Speicher für Zwischenergebnisse
#0089     Speicher für Zwischenergebnisse
#008A     Speicher für Zwischenergebnisse
#008B-    Speicher für Ergebnisse bei Berechnungen
#008E
#008F-    Akkumulator für Berechnungen
#0093
#0094/5   Zeiger auf Directory-Puffer; enthält #05/02
#0096     Kommando vom IEEE-Bus; hier unbenutzt
#0098     Bitzähler für seriellen Bus
#0099/A   Buffer-Pointer für Puffer 0; steht auf #0300
#009B/C   Buffer-Pointer für Puffer 1; steht auf #0400
#009D/E   Buffer-Pointer für Puffer 2; steht auf #0500
#009F/0   Buffer-Pointer für Puffer 3; steht auf #0600
#00A1/2   Buffer-Pointer für Puffer 4; steht auf #0700
          Alle diese Pointer werden durch den B-P-Befehl ver-
          ändert!
#00A3/4   Zeiger auf nächstes Zeichen im INPUT-BUFFER (#0200)
#00A5/6   Zeiger auf nächstes Zeichen im ERROR-BUFFER (#02D6)
#00A7-    Tabelle: enthält für jeden aktiven Puffer die entspre-
#00AD     chende Kanalnummer. Kanalnummer = #FF, wenn Puffer
          unbenutzt.
#00AE-    Tabelle: enthält für jeden aktiven Puffer die entspre-
#00B4     chende Kanalnummer. Kanalnummer = #FF, wenn Puffer
          unbenutzt.
#00B5-    Tabelle der Lo-Bytes der Recordnummern für jeden Puffer
#00BA
#00BB-    Tabelle der Hi-Bytes der Recordnummern für jeden Puffer
#00C0
#00C1-    Tabelle der nächsten zu bearbeitenden Recordnummern
#00C6     für jeden Puffer
#00C7-    Tabelle der Recordlängen für jeden Puffer
#00CC
#00CD-    Tabelle der Side-Sektoren für jeden Puffer
#00D2
#00E2-    Standardwerte für Laufwerk; hier alle 0
#00E6
#00E7-    Tabelle der Filetypen
#00EB
#00EC-    Kanal Filetyp
#00F1
#00F2-    Kanalstatus
#00F7
#00F8     Zwischenspeicher für EOI
#00F9     Aktuelle Puffernummer für Befehlscode
#0101     Formatkennzeichen von Spur 18 Sektor 0
#0104-    Bereich des Hardware-Stack; nicht benutzbar
#0145
#0200-    INPUT-BUFFER; hier werden alle Befehlsstrings vom Com-
#0229     puter zwischengespeichert und nach Syntaxprüfung aus
          geführt
#022A     Codenummer des auszuführenden Befehls
#022B-    Kanaltabelle; diese Tabelle enthält für jede mögliche
#023E-    Aktuelles Datenbyte für jeden Kanal; Belegung der
#0243     Adressen wie bei der Kanalstatustabelle (#022B)
#0244-    Tabelle der Zeiger auf das letzte aktuelle Zeichen in
#0249     jedem, für den Kanal zuständigen, Pufferspeicher
#024A     Gerade behandelter Filetyp
#024B     Länge des Befehlsstrings
#024C     Zwischenspeicher für Sekundäradresse
#024D     Zwischenspeicher für Befehlscode
#024E     Arbeitsspeicher beim Suchen des nächsten Sektors
#024F/0   Pufferbelegungsspeicher; 1 = Puffer belegt
#0253     Flag für Directory-Eintrag gefunden
#0254     Flag für $-Befehl zum Listen des Directory
#0255     Flag für Befehlsausführung (<>#00, wenn Befehl anliegt)
#0257     Nummer des letzten benutzten Puffers
#0258     Recordlänge
#0259     Side-Sector Spur
#025A     Side-Sector Sektor
#025B-    Tabelle; enthält den letzten Befehlscode der Puffer
#025F
#0260-    Sektornummern der Directoryeinträge in den Puffern
#0265
#0266-    Zeiger auf die Directoryeinträge in den Puffern
#026B
#026D     Flag für LED Blinken bei Fehler
#026E     Nummer des letzten aktiven Laufwerks
#026F     Nummer des letzten bearbeitenden Sektors
#0270     aktueller Schreibkanal
#0271     aktueller Lesekanal
#0274     Länge des Befehlsstrings im INPUT-BUFFER
#027A-    Tabelle der Zeiger auf die Filenamen
#027F
#0280-    Spurnummern der Files für den aktuellen Puffer
#0284
#0285-    Sektornummern der Files für den aktuellen Puffer
#0289
#028A     Joker (*) Flag
#028E     Standardwert für die Nummer des Laufwerks
#028F     Flag für Fileeintrag im Directory gefunden
#0290     Sektornummer des aktuellen Directory Sektors
#0291     Sektornummer des ersten Directoryeintrags
#0292     Zeiger auf ersten gültigen Directoryeintrag
#0293     Zeigt letzten Block an; enthält dann 0
#0294     Aktueller Pufferzeiger
#0295     Zähler für Fileeinträge
#0297     Betriebsart des aktuellen Files (Lesen/Schreiben)
#029D/E   Spurnummer der BAM
#02A1-    Zwischenspeicher für BAM Eintragungen
#02B0
#02B1-    Puffer für Directory
#02D4
#02D5-    ERROR-BUFFER; enthält auszugebende Fehlermeldung
#02F8
#02FA     Lo-Byte der Anzahl der freien Blocks auf Diskette
#02FC     Hi-Byte der Anzahl der freien Blocks auf Diskette
#0300-    Puffer 0
#03FF
#0400-    Puffer 1
#04FF
#0500-    Puffer 2
#05FF
#0600-    Puffer 3
#06FF
#0700-    Puffer 4 (enthält normalerweise die BAM)
#07FF
#0800-    Nicht mit RAM belegt
#FFFF
```

```
100 REM AUTO-'&'-MAKER                           <106>
110 REM --------------                           <115>
120 REM                                          <007>
130 REM  03.11.84.  BORIS SCHNEIDER              <224>
140 :                                            <198>
150 :                                            <208>
160 REM INITIALISIERUNG                          <159>
170 INPUT"STARTADRESSE DES &-FILES";SA           <121>
180 INPUT"NAME DES &-FILES";NA$                  <046>
190 IF LEN(NA$)>15 THEN 180NA$                   <026>
200 OPEN 1,8,2,"&"+NA$+",U,W"                    <063>
210 DIM X(256)                                   <158>
220 PRINT"BITTE GEBEN SIE JETZT IHRE DATEN EIN"
                          <116>
230 PRINT"ABSCHLUSS MIT -1!"                     <212>
240 :                                            <042>
250 REM DATENEINGABE UND TEST AUF                <227>
260 REM UEBERLAUF                                <047>
270 Y=1                                          <075>
280 INPUT X(Y)                                   <160>
290 IF X(Y)<0 THEN Y=Y-1:GOTO 350                <213>
300 PR=PR+X(Y):IF PR>255 THEN PR=PR-255          <103>
305 Y=Y+1:IF Y>254 THEN 350                      <026>
310 GOTO 280                                     <090>
320 :                                            <123>
330 REM ABSPEICHERN DER VORHANDENEN              <017>
340 REM DATEN IN DAS USR-FILE                    <006>
350 SH=INT(SA/256)                               <144>
360 SL=SA-256*SH                                 <221>
370 PR=PR+SH+SL+Y                                <250>
380 PRINT#1,CHR$(SL);CHR$(SH);                   <082>
390 PRINT#1,CHR$(Y);                             <040>
400 FOR I=1 TO Y                                 <059>
410 PRINT#1,CHR$(X(I));                          <213>
420 NEXT                                         <039>
430 PR=PR-(255*INT(PR/256))                      <219>
440 PRINT#1,CHR$(PR);                            <163>
450 IF X(Y+1)<0 THEN GOTO 470                    <217>
460 SA=SA+Y:PR=0:GOTO 270                        <196>
470 CLOSE 1                                      <168>
```

Listing 4. Komfortable »&-Files« erzeugen

zeichnet sein. Sie enthalten als erstes Zeichen im Filenamen das Zeichen »&«. Soll also zum Beispiel ein File mit dem Namen »Test« als Autostartprogramm in der Floppy ausgeführt werden, so geben Sie diesem File den Namen »&Test« und starten Sie es danach mit:
OPEN1,8,15,"&TEST"

Haben Sie nur ein einziges Autostartfile auf Diskette, so können Sie es auch nur mit »&« abspeichern und ebenso mit
OPEN1,8,15,"&" starten.

Leider erwartet die Floppy von Autostartfiles eine spezielle Syntax, die in Tabelle 3 zu sehen ist.

Als Listing 3 haben wir noch einmal unser LED-Testprogramm; nur wird diese Routine durch das Basic-Programm als &-File auf Diskette geschrieben und kann danach durch den schon erwähnten Befehl direkt von Diskette in den Pufferspeicher geschrieben und dort gestartet werden.

Zu Tabelle 3 noch einige Anmerkungen:

Zuerst muß die Startadresse des Programms im Pufferspeicher der Floppy in das File geschrieben werden. Danach folgt die Anzahl der Bytes im Programm. Jetzt werden die Programmbytes abgespeichert, und schließlich folgt noch eine Prüfsumme, die sich wie folgt errechnet:

Es werden alle Bytes des Programms addiert und zum Ergebnis noch die zwei Bytes der Startadresse und die Anzahl der Bytes im Programm hinzugezählt. Dieses Ergebnis ist als Integerzahl zu verstehen und besteht also aus einem niederwertigen (LO) und einem höherwertigen (HI) Byte. Das niederwertige Byte ist die Prüfsumme, zu der noch der Übertrag im höherwertigen Byte addiert werden muß. Diese Berechnung klingt kompliziert; ist es aber nicht. In Listing 4 wird Ihnen diese Rechnerei abgenommen. Die allgemeine Formel hier noch einmal:

HB = INT(SUMME/256)
LB = SUMME—HB*256

dabei bedeuten:

HB — das höherwertige Byte
LB — das niederwertige Byte
SUMME — die Gesamtsumme der Programmbytes

Achtung: Die Übertragsberechnung muß nach jedem neu dazugezählten Wert erfolgen, da das Endergebnis kleiner als 256 sein muß! Wie Sie sehen, ist das Anlegen eines &-Files nicht ganz einfach. Bisher wurde diese Fileart fast nur von Profis zum Programmschutz angewandt, da sie, wie schon erwähnt, nahezu unbekannt war.

Zu erwähnen wären noch zwei seltsame Fehlermeldungen der Floppy:

»OVERFLOW IN RECORD« erscheint, wenn die Anzahl der tatsächlichen Bytes mit der Angabe nicht übereinstimmt.

»RECORD NOT PRESENT« erscheint, wenn die Prüfsumme nicht stimmt.

Da wir stets darum bemüht sind, Ihnen die Arbeit mit der Floppy so angenehm wie möglich zu machen, haben wir unserem Artikel noch Listing 4 beigefügt. Es handelt sich hier um ein Programm, das es Ihnen gestattet, auf einfachste Weise &-Files zu erstellen. Diese können sogar länger als 256 Byte sein, da das Programm dann automatisch eine Prüfsumme und die Anschlußadresse einfügt. Ununterbrochene &-Files, die länger als 256 Zeichen sind, kann es ja nicht geben, da die Anzahl der Programmbytes im File nur in einem Byte abgespeichert wird.

Mit dieser neuen Fileart wollen wir Sie für dieses Mal entlassen. Ruhen Sie sich für die nächste Folge aus. Wir werden dann auf die Technik der Jobschleifenprogrammierung eingehen, die Ihnen eine Fülle von Anwendungen eröffnen wird.

(K. Schramm/B. Schneider/gk)

DISKCONTROLLER (DC)

VIA 6522, $ 1800, PORT B

Bit #	Bedeutung
0	DATA IN
1	DATA OUT
2	CLOCK IN
3	CLOCK OUT
4	ATN OUT
5 } 6 }	GERÄTENUMMER
7	ATN IN (CB 2)

BUSCONTROLLER (BC)

VIA 6522, $1C00, PORT B

Bit #	Bedeutung
0	Steppermotor für Laufwerk 1 (n.v.)
1	Steppermotor für Laufwerk 0
2	Laufwerksmotor
3	LED am Laufwerk (rot)
4	Schreibschutzkennung
5	Bitsynchronisation für DC bei den vier
6	Spurbereichen
7	SYNC-Signal

Tabelle 2. Belegung der beiden Controll-Ports der 1541

Byte	Bedeutung
1-2	Startadresse in der 1541 im HI/LO-Format
3	Anzahl der folgenden Programmbytes
4-N	Programm
N+1	Prüfsumme
N+2	Hier kann bei längeren Programmen ein weiterer Teil eingefügt werden. Format: wieder bei Byte 1 beginnend.

Tabelle 3. Aufbau eines &-Files. In dieser Tabelle sind die Linker- beziehungsweise Endekennzeichen, die in den ersten beiden Bytes eines Datenblocks stehen, nicht enthalten, da sie beim Öffnen und Beschreiben eines &-Files automatisch gesetzt werden.



Uns erreicht eine Unmenge an Zuschriften von Lesern, die einen Anschluß an einen Club in ihrer Nähe suchen. Soweit Clubs in dieser Stadt oder dem Postleitzahlengebiet bekannt sind, geben wir die Adressen natürlich gerne weiter. Doch es wird sicherlich noch genügend andere Commodore-Besitzer geben, die in einem Club mitmachen wollen.



sich bei uns zu melden. Wichtig sind dabei neben der Adresse auch die Schwerpunkte, mit denen sich der Club befaßt. Seien dies nun der Erfahrungs- oder Programmaustauch, die DFÜ, die Hardware oder eine eigene Clubzeitschrift. Um möglichst alle Clubs in der geplanten Übersicht veröffentlichen zu können, sollten sie sich bei Ihren Angaben auf das Notwendigste beschränken. Also Adresse und vier oder fünf Stichpunkte, etwa nach diesem Schema:

C 64 User-Club POKE-Freunde, Basic-Str. 1, 1024 Commodorestadt,
Clubtreffen, monatliche Zeitschrift, Softwarebibliothek, Hardware, Funker, DFÜ, etc.

Diese Infos schicken Sie bitte an: Markt & Technik Verlag AG, Redaktion 64'er, Stichwort: Club, Hans-Pinsel-Str. 2, 8013 Haar bei München.

Einmal im Monat gibt es die SUPERCHANCE

Diese nicht einmalige Gelegenheit sollten Sie nutzen. Wie? Schicken Sie uns Ihr bestes, selbst erstelltes Programm. Bei der Art des Programms sind wir nicht wählerisch.

Sie haben ein sehr gutes (Schieß-, Knobel-, Denk-, Action-, Abenteuer-)Spiel geschrieben: einschicken!

Sie verfügen über ein komfortables Disketten-Kopier-(Sortier) Programm mit einigen außergewöhnlichen Leistungsmerkmalen: einschicken!

Sie haben das Basic um einige sinnvolle Befehle erweitert: einschicken!

Sie arbeiten mit einem selbsterstellten Textverarbeitungsprogramm, einer eigenen Tabellenkalkulation, einem semiprofessionellen Datenverwaltungsprogramm: einschicken!

Sie zeichnen und konstruieren mit einem selbsterstellten Programm in hochauflösender Grafik: einschicken!

Wir freuen uns über jeden Beitrag und honorieren mit bis zu

2000 Mark

für das Listing des Monats

Aus den besten Listings, die veröffentlicht werden, sucht die 64'er-Redaktion einmal im Monat das »Listing des Monats« aus. Alle Listings, die im 64'er abgedruckt sind, werden mit 100 bis 300 Mark honoriert. Die genaue Vorgehensweise beim Einsenden von Listings ist in dem Beitrag »Wie schicke ich meine Programme ein?« in verschiedenen Ausgaben beschrieben.

Schicken Sie Ihr Listing an: Redaktion 64'er, Superchance: Listing des Monats, Hans-Pinsel-Str. 2, 8013 Haar bei München.



Die Diskette für eine Ausgabe kostet **29,90** Mark. Sie werden bei einigen Disketten bestimmte Programme vermissen. Deren Autoren konnten sich nicht entschließen, ihr Programm im Rahmen des Leserservice für eine Verbreitung auf Datenträger freizugeben. Bei den Ausgaben 5 und 6 können noch Kassetten (VC ...) bestellt werden. Zu den Programmen sind immer die Seitenzahlen anzugeben, unter der Sie die Beschreibungen in der entsprechenden Ausgabe finden können. Der Diskette liegen also keinerlei Informationen bei. Lesen Sie daher aufmerksam die Anleitung (ob SYS-Befehle nötig sind, in welcher Reihenfolge geladen werden muß, eventuelle Sprach- oder Speichererweiterungen und ähnliches mehr) in dem jeweiligen Artikel nach. Aus Aktualitätsgründen wird jeweils die abgedruckte Version angeboten. **Eventuelle systematische Fehler, die sich noch im Programm befinden können, müssen von Ihnen selbst, nach Studium des Druckfehlerteufelchens, korrigiert werden.**

Ausgabe 1/85

Bestell-Nr. L 6 8501A DM 29,90*

Commodore 64
Checksummer 64
Handballtrainer (AdM)
SMON Teil 3
Hi-Eddi (LdM)
Hypra-Load mal vier
Tips und Tricks
Provic 64
Eingabe (UPB)

VC 20
Checksummer VC 20

Ausgabe 12/84

Bestell-Nr. CB 022 DM 29,90*

Commodore 64
Synthesizer (AdM) S.51
SMON (2. Teil) S.60
3D-Vier gewinnt S.96
Trace S.76
Stringy S.88
Lader S.92
Auto S.84
Listschutz S.85
Simons Axo (SB) S.64
Kreuzworträtsel S.150

VC 20
Mathematikal Basic (8K>) (LdM) S.55
Fast Tape S.80

Ausgabe 11/84

Bestell-Nr. CB 020 DM 29,90*

Commodore 64
Turtle Grafik (LdM) S.48
Schachmeister (AdM) S.50
SMON (1. Teil) S.59
Floppykurs S.117
FPLOT-Befehlserweiterung S.73
Get Koala pic S.66
Interrupttechnik S.84
Exsort (UPB) S.154
Einzeiler S.158
Simons Basic Befehlserweiterung (SB) S.90

VC 20
Pseudosprites (8K) S.76
Laterna Magica (8K) S.68
Betriebssystem-Erweiterung (24K>) S.88
Supergrafik (GV) S.71
VC 20-Kurs (GV>) S.126

Ausgabe 10/84

Bestell-Nr. CB 019 DM 29,90*

Commodore 64
Finanzmathematik (AdM) S.68
Hypra-Load (LdM) S.67
Hardcopy Compact 2 S.86
Hardcopy MPS 801 S.82
Hardcopy VC 1526 neu S.83
Hardcopy Gemini-10X S.85
Hardcopy FX-80 S.88
Hardcopy VC 1520 farbig S.84
Apocalypse now S.106
Supercopy S.102
Disk-Dump S.96
Diskettenorganisation S.97
User-Port-Tastatur S.92
Maske (UPB) S.172

VC 20
Epedemic S.112
Video-Vorspann S.81

Ausgabe 9/84

Bestell-Nr. CB 014 DM 29,90*

Commodore 64
Indexsequentielle Adreßdatei S.54
Spring Vogel (LdM) S.68
Orgel/Synthesizer (AdM) S.70
Sprite Aid + S.89
Screen Change S.94
List-Stop S.97
Renew, Datawandler S.102
Synthetische suchen S.104
Geregelter Zahlungsverkehr S.164

VC 20
Schiebung (GV>) S.77
Deuzei (8K>) S.79
Hardcopy 1520 (GV>) S.87
RS232-Interface (GV>) S.100
Datawandler (GV>) S.102

Ausgabe 8/84

Bestell-Nr CB 013 DM 29,90*

Commodore 64
Castle of Doom S.66
Pac-Boy S.89
Kopplung S.73
User-Port-Display S.97
RS232-Test S.77
View BAM S.99
Görlitz Hardcopy S.83
Milchvieh S.156

VC 20
Kudiplo (3K) S.86
Print at Restore n (GV) S.101

Ausgabe 7/84

Bestell-Nr. CB 017 DM 29,90*

Commodore 64
Terminalprogramm S.24
Softwarekatalog S.72
Russvok (SB) S.76
Crown No. 1 S.80
Space Invaders S.81
1520 Hardcopy S.108
Centronics Interface S.110
Kurvendiskussion S.116
Copy Rel. Files S.132
Autostart S.138
Strubs (OP u. QP) S.154

VC 20
Rätsel S.122

Ausgabe 6/84

Bestell-Nr. CB 018 DM 29,90*

Commodore 64
Lehrerkalender S.64
Morsetrainer S.72
Supervoc S.69
Grafische Darstellung (SB) S.82
Hot Wheels S.92

VC 20
Bestell-Nr. VC 008 DM 29,90*
Movemaster (8K) S.78
Ghost Manor (GV) S.104
Logic Disass. (3K>) S.108
Underground (LdM 16K) s.120

Ausgabe 5/84

Bestell-Nr. CB 016 DM 29,90*

Commodore 64
Adreß-&Telefonregister S.64
Fahrsimulator S.82
Schatzsucher (LdM) S.90

VC 20
Bestell-Nr. VC 007 DM 29,90*
Relative Datei (8K) S.69
Schmatzer (GV) S.76
3D-Grafik (8K) S.78
Rallye (28K) S.128

* Alle Preise inklusive Mehrwertsteuer. Der Versand erfolgt mit offener Rechnung zuzüglich Porto und Verpackung.

Fehlende Hefte erhalten Sie bei: Markt & Technik Vertrieb 64'er Hans-Pinsel-Str. 2, 8013 Haar

Bedeutung der Abkürzungen

*LdM = Listing des Monats
*AdM = Anwendung des Monats
*SB = Simons Basic
*GV = Grundversion
*GV> = alle Speicherversionen können verwendet werden (einschließlich GV)
*3K = 3-KByte-Speichererweiterung wird benötigt
*8K> = Speichererweiterung größer als 8 KByte wird benötigt
*UPB = Unterprogrammbibliothek

Bestellungen richten Sie bitte an: M&T Buchverlag, Hans-Pinsel-Str. 2, 8013 Haar bei München

500 Mark für formatierte Eingabe

Schreiben Sie ein Programm, dann besteht es in der Regel aus drei Teilen: Eingabe, Verarbeitung und Ausgabe von Daten. Es sind also diese Teile, die immer wieder programmiert werden müssen. Und deshalb hat dieses Mal ein Unterprogramm den Wettbewerb gewonnen, das eine wichtige und universelle Eingabe-Routine zur Verfügung stellt.

Jeder Programmierer steht bei einem neuen Programm vor der gleichen Frage: Wie soll meine Eingabe aussehen? Begnügt man sich mit dem vorhandenen Wortschatz des Commodore-Basic, ist es schwierig, eine sichere Eingabe zu erhalten. Andere Basic-Versionen haben entsprechende Befehle. Was also liegt näher, als sich eine Eingabe-Routine selbst zu programmieren, und zwar eine, die man immer wieder verwenden kann? Doch da beginnt auch schon das Problem. Es ist gar nicht so einfach, Standard-Unterprogramme zu entwickeln. So ein Programm muß folgende Forderungen erfüllen:

1. Es muß fehlerfrei sein. Das heißt, ganz gleich, welche Taste bei einer Eingabe gedrückt wird, das Programm darf niemals mit einer Fehlermeldung (oder noch schlimmer: ohne Meldung) abstürzen.
2. Es muß flexibel sein. Man muß das Unterprogramm in jedem möglichen Programm einsetzen können, unabhängig von der sonstigen Aufgabe. Die wichtigsten Eingaben sind: nur numerische, also alle Ziffern von 0 bis 9; für kaufmännische und ähnliche Probleme muß wahlweise zusätzlich noch die Eingabe eines Dezimalpunktes möglich sein; und Text, also alle Buchstaben, Sonderzeichen und Ziffern. Oft wird eine bestimmte Eingabelänge gefordert und darf nicht überschritten werden, auch das ist wichtig. Und damit die einzugebenden Werte auch an der richtigen Stelle stehen, muß der Cursor positioniert werden können.

Das folgende Unterprogramm erfüllt diese Forderungen und läßt sich in jedem Basic-Programm einsetzen. Die Bedienung des Programms können Sie den REM-Zeilen des Listings entnehmen.

(Rolf Hilchner/gk)

```
1 REM" ┌──────────────────────────────────┐
2 REM" |   UP  -  FORMATIERTE EINGABE     |
3 REM" └──────────────────────────────────┘
4 REM" ┌──────────────────────────────────┐
5 REM" |   Q1   =  ZEILE DER EINGABE      |
6 REM" |   Q2   =  SPALTE DER EINGABE     |
7 REM" |   Q3   =  MAX. LAENGE D. EING.   |
8 REM" |   Q4   =  1 ← LEEREINGABE NICHT  |
9 REM" |             ERLAUBT              |
10 REM"|   Q5   =  1 ← NUR NUMERISCHE     |
11 REM"|             EINGABE ERLAUBT      |
12 REM"|   Q6   =  1 ← BEI NUMERISCHER    |
13 REM"|             EINGABE IST EIN      |
14 REM"|             PUNKT ERLAUBT        |
15 REM"|                                  |
16 REM"|   Y3$ =  MUSS EINMAL AM ANFANG   |
17 REM"|          DES HAUPTPROGRAMMES     |
18 REM"|          DEFINIERT WERDEN. EIN   |
19 REM"|          PUNKTESTRING, DER DIE   |
20 REM"|          LAENGE DER LAENGSTEN    |
21 REM"|          EINGABE IM HAUPTPROG.   |
22 REM"|          HAT. ZUSAETZLICH MUSS   |
23 REM"|          AM ENDE DES STRINGS     |
24 REM"|          DAS ZEICHEN '←'         |
25 REM"|          STEHEN. BEISPIEL :      |
26 REM"|          Y3$=''..............←'' |
27 REM"|                                  |
28 REM"|   Y4$ =  WIE 'Y3$', JEDOCH       |
29 REM"|          WERDEN STATT PUNKTE     |
30 REM"|          SPACES GESETZT. DAS     |
31 REM"|          ZEICHEN '←' ENTFAELLT   |
32 REM"|          BEISPIEL :              |
33 REM"|          Y4$=''              ''  |
34 REM"└──────────────────────────────────┘
35 REM"┌──────────────────────────────────┐
36 REM"|   II   =  LAUFVARIABLE           |
37 REM"|   JJ   =  LAUFVARIABLE           |
38 REM"|   Q7   =  ZAEHLER, DER ANGIBT,   |
39 REM"|           AN WELCHER STELLE EIN  |
40 REM"|           PUNKT GESETZT WURDE    |
41 REM"|                                  |
42 REM"|   Y1$  =  INHALT DER EINGABE     |
43 REM"|           WIRD VOM UNTERPROG.    |
44 REM"|           AN DAS HAUPTPROG.      |
45 REM"|           UEBERGEBEN.            |
46 REM"|                                  |
47 REM"|   Y2$  =  ENTHAELT DAS JEWEILS   |
48 REM"|           EINGEGEBENE ZEICHEN    |
49 REM"└──────────────────────────────────┘
50 REM"┌──────────────────────────────────┐
51 REM"|  FORMATIERTE EINGABE, BEI DER    |
52 REM"|  JEDES EINGEGEBENE ZEICHEN       |
53 REM"|  UEBERPRUEFT UND FALLS NOETIG    |
54 REM"|  ZURUECKGEWIESEN WERDEN KANN.    |
55 REM"|  DIE ANGEGEBENEN PARAMETER       |
56 REM"|  MUESSEN ZUM TEIL (Q1,Q2,Q3,     |
57 REM"|  Y3$,Y4$) EINGEGEBEN WERDEN,     |
58 REM"|  ODER KOENNEN BEI BEDARF         |
59 REM"|  UEBERGEBEN WERDEN (Q4,Q5,Q6)    |
60 REM"|  VOR DEM RUECKSPRUNG INS         |
61 REM"|  HAUPTPROGRAMM WERDEN ALLE       |
62 REM"|  PARAMETER AUF NULL GESETZT.     |
63 REM"|  DAS ERGEBNIS DER EINGABE-       |
64 REM"|  ROUTINE, DIE EINGABE WIRD IN    |
65 REM"|  DER VARIABLEN 'Y1$' AN DAS      |
66 REM"|  HAUPTPROGRAMM UEBERGEBEN.       |
67 REM"└──────────────────────────────────┘
68 REM
69 REM
70 REM
100 REM  ****************************
102 REM  *** FORMATIERTE EINGABE ***
104 REM  ***         VON          ***
106 REM  ***    ROLF HILCHNER     ***
108 REM  ***   RHEYDTER STR.48    ***
109 REM  ***    4040 NEUSS 1      ***
```

```
110 REM    ****************************
120 POKE 650,255:Y1$="":Q2=Q2-1:POKE 214,Q1
   :POKE 211,Q2:PRINT"{UP}T";RIGHT$(Y3$,Q3)
                       <071>
130 FOR II=1 TO Q3+1                    <123>
140 GET Y2$:IF Y2$=""THEN 140           <130>
150 IF ASC(Y2$)=20 AND II>1 THEN Y1$=LEFT$(Y1$,
   LEN(Y1$)-1):II=II-2:GOTO 250         <253>
160 IF ASC(Y2$)=13 AND II=1 AND Q4=1 THEN GOSUB
    280:GOTO 140                        <104>
170 IF ASC(Y2$)<>13 AND II=Q3+1 THEN GOSUB 280
   :GOTO 140                            <059>
180 IF ASC(Y2$)=13 GOTO 260             <060>
190 IF ASC(Y2$)<32 OR ASC(Y2$)>93 THEN GOSUB 280
   :GOTO 140                            <224>
200 IF Q5=1 AND ASC(Y2$)=45 AND II=1 GOTO 240
                       <143>
210 IF II<=Q7 THEN Q7=0:Q6=1            <145>
220 IF Q5=1 AND ASC(Y2$)=46 AND Q6=1 THEN Q6=0:
   :Q7=II:GOTO 240                      <035>
230 IF Q5=1 AND ASC(Y2$)<48 OR Q5=1 AND ASC(Y2$
   )>57 THEN GOSUB 280:GOTO 140         <063>
240 Y1$=Y1$+Y2$                         <087>
250 POKE 214,Q1:POKE 211,Q2:PRINT"{UP}";
   LEFT$(Y1$+"T"+Y3$,Q3)+"←";:NEXT II   <004>
260 POKE 214,Q1:POKE 211,Q2:PRINT"{UP}";
   Y1$+LEFT$(Y4$,Q3-LEN(Y1$)+1)         <242>
270 Q1=0:Q2=0:Q3=0:Q4=0:Q5=0:Q6=0:Q7=0
   :POKE 650,0:RETURN:'<--- AUSGANG AUS UP <117>
280 POKE 54296,15:POKE 54277,6:POKE 54278,0
   :POKE 54275,8:POKE 54274,0           <046>
290 POKE 54273,92:POKE 54272,237:POKE 54276,65
   :FOR JJ=1 TO 150:NEXT JJ:POKE 54276,0  <084>
300 RETURN:'<--- AUSGANG AUS PIEP-UP    <061>
310 REM"┌──────────────────────────────┐
320 REM"| BEISPIEL 1 FUER DEN AUFRUF :  |
330 REM"|                               |
340 REM"| 10 Y3$=''..........←'':Y4$='' |
350 REM"|                  ''           |
360 REM"| 20 PRINT''□'':POKE214,5:POKE  |
370 REM"|    211,15:PRINT''□NAME :''    |
380 REM"| 30 Q1=5:Q2=23:Q3=10:GOSUB 100 |
390 REM"| 40 PRINT:PRINT''SIE HEISSEN'' |
400 REM"|    ;Y1$:END                   |
410 REM"|                               |
420 REM"| BEISPIEL 2 FUER DEN AUFRUF :  |
430 REM"|                               |
440 REM"| 10 Y3$=''..........←'':Y4$='' |
460 REM"| 20 PRINT''□'':POKE214,5:POKE  |
470 REM"|    211,15:PRINT''□ALTER :''   |
480 REM"| 30 Q1=5:Q2=24:Q3=10:Q4=1:     |
490 REM"|    Q5=1:GOSUB 100             |
500 REM"| 40 PRINT:PRINT''SIE SIND'';   |
510 REM"|    VAL(Y1$);''JAHRE ALT'':END |
520 REM"|                               |
530 REM"| HINWEIS :                     |
540 REM"| ─────────                     |
550 REM"| POKE 214,Y BEWIRKT, DASS DER  |
560 REM"| CURSOR IN DIE ZEILE Y SPRINGT |
570 REM"|                               |
580 REM"| POKE 211,X BEWIRKT, DASS DER  |
590 REM"| CURSOR AN D. SPALTE X SPRINGT |
600 REM"└──────────────────────────────┘

READY.
```

Formatierte Eingabe

Einzeiler-Wettbewerb: Die nächsten 14

Wieder haben wir die interessantesten Einzeiler für Sie herausgesucht. Darunter sind sowohl nützliche als auch witzige oder lehrreiche. Sogar ein Adventure ist darunter. Alle Veröffentlichungen werden mit 50 Mark und einer Diskette mit allen Programmen dieser Ausgabe belohnt.

Wenn man einigen Einsendern Glauben schenken will — und es besteht kein Grund, das nicht zu tun —, so sind ihre Einzeiler keineswegs schnell dahingeschriebene Programme, sondern oft das Ergebnis von tage-, ja wochenlangen Experimenten und Versuchen. Bei manchen anderen allerdings ist die programmiertechnische Umsetzung einer Idee nicht das Entscheidungskriterium gewesen, sondern vielmehr die Idee selbst. Sie werden auch sehen, daß bei einem Einzeiler weder das Programm noch die Idee besonders originell ist. Aber seine Programmbeschreibung ist so überzeugend, daß sie Ihnen nicht vorenthalten sein soll. Doch nun viel Spaß und hoffentlich einige »Aha«-Erlebnisse.

Umwandlung beliebiger Zahlensysteme (VC 20/C 64)

Die beiden Einzeiler dienen zum Umrechnen zwischen Dezimalzahlen und Zahlen beliebiger Basis. Kombiniert bilden beide eine Umwandlung zwischen verschiedensten Zahlensystemen. Beide Zeilen können sowohl als Unterroutine (mit RETURN) wie auch als Teil eines größeren Programms stehen.

a) Umwandlung dezimal/beliebig

Die Routine wandelt eine Dezimalzahl beliebiger Größe in der Variablen D in eine Zahl der Basis um, die in der Variablen B angegeben ist. Das Ergebnis steht in Z$. Zuerst wird Z$ gelöscht. Dann wird eine Dummy-Schleife eröffnet, die nur einen einzigen Durchlauf zu haben scheint (0 bis 0). Der Befehl wird dazu genutzt, später wieder mitten in die Zeile einspringen zu können. Bei jeder Stelle wird D durch die Basis B geteilt und dadurch die unterste Ziffer abgeschnitten. Die jeweils niederwertigste Stelle ist der ganzzahlige Rest dieser Division und steht in S. S wird nun in den ASCII-Code umgerechnet, indem zur Zahl S 45 addiert wird. Ist S eine Ziffer von 0 bis 9, so nimmt der Term (S < 10) den Wert —1 an und es wird 7 subtrahiert (siehe ASCII-Tabellen). Der Code wird durch den CHR$-Befehl in einen String gewandelt und vorne an Z$ angehängt. Die letzte (höchstwertige) Ziffer ist erreicht, wenn D < 1 ist (nächste Stelle = 0). Die Schleifenvariable P wird auf —D gesetzt. Beim darauffolgenden NEXT-Kommando wird P um den STEP (hier 1) erhöht und mit dem Endwert des FOR-Befehls (hier 0) verglichen. Wenn D noch > = 1 ist, ist P + 1 < = 0, die Endbedingung ist noch nicht erreicht und es wird zu dem Statement nach dem FOR-Befehl gesprungen.

b) Umwandlung beliebig/dezimal

Diese Routine wandelt eine beliebige große Zahl der von der Variablen B angegebenen Basis in eine Dezimalzahl. Die zu wandelnde Zahl muß in Z$ stehen; die Variable D enthält das Ergebnis. Am Anfang wird die Variable D auf Null gesetzt. Die Schleifenvariable S des darauffolgenden FOR-Befehls dient als Zeiger auf die einzelnen Stellen von Z$. Diese werden nun in den ASCII-Code gewandelt, der Code für Null (48) wird subtrahiert und das Ergebnis in H zwischengespeichert. Die schon umgewandelten Stellen (in D) werden zunächst durch Multiplikation mit B um eine Potenz dieser Basis erhöht, um dann die aktuelle Stelle (H) zu addieren. Ist

H>9 (Darstellung durch einen Buchstaben) müssen aufgrund des ASCII-Codes noch sieben abgezogen werden (Term (H>9) wird −1). Nach der niederwertigsten Stelle ist die Schleife beendet.

(Martin + Hartmut Sprave)

```
10 z$="":forp=0to0:d=d/b:s=(d-int(d))*b:
z$=chr$(55+s+7*(s<10))+z$:p=-d:next
11 :
12 :
13 :
20 d=0:fors=1tolen(z$):h=asc(mid$(z$,s))
-48:d=d*b+h+7*(h>9):next
21 :
22 :
23 :
30 rem zahlenumwandelung:
40 rem in 10 dez in beliebig
50 rem in 20 beliebig in dez
60 :
```

Scrollen in x-Richtung (C 64)

FORG=0TO3:NEXT ist eine Verzögerungsschleife; hier kann die Geschwindigkeit des Scrollers eingestellt werden. Zu beachten ist, daß alle Befehle in abgekürzter Schreibweise eingegeben werden müssen.

(Hans-Peter Harmann)

```
10 fort=1to7:poke53270,t:forg=0to3:next:
next:onagoto10:fory=1024to2023:pokey,194
:next:a=1:goto10
20 rem
```

Das abwechslungsreiche Programm (C 64)

Liebe Redaktionäre, endlich ein Wettbewerb, an dem auch ich als Novize mich beteiligen kann. Meine Computer- und Programmierkenntnisse sind mittlerweile schon weit fortgeschritten (die 1-Prozent-Hürde werde ich demnächst überspringen), so daß ich mich berufen glaube, auch etwas zum Besten zu geben.

Ich stellte mir selbst hohe Anforderungen:
1. Das Programm sollte abwechslungsreich sein
2. Es sollte Bewegung im Spiel sein
3. Die Farbe durfte nicht fehlen.

Bitte das Programm sorgfältig abschreiben, um eine langwierige Fehlersuche zu vermeiden.

Variablenliste:
A$ = Stunden von TI$
B$ = Minuten von TI$
C$ = Sekunden von TI$

Vor Eingabe des Programms muß selbstverständlich TI$ auf die aktuelle Uhrzeit eingestellt werden. Die Zeilennummer 11 fristet für gewöhnlich ein Schattendasein hinter der 10. Deshalb habe ich aus Mitleid die 11 gewählt. Nachdem Sie den Bildschirm durch CLR/Home gereinigt haben, starten Sie das Programm. Sie sehen, daß meine Anforderungen erfüllt wurden.

1. Das Programm ist abwechslungsreich (Wiederholungen treten höchstens nach 24 Stunden auf)
2. Es bewegt sich was
3. Es ist Farbe im Spiel

Zu 3. Das Programm wurde so gestaltet, daß der Anwender leicht die Farbe ändern kann.

PS: Das Ausschalten des Computers bringt das Programm zum Abstürzen!!

P.P.S: Freundlich zugedachte Geldspenden werden nur auf Antrag angenommen.

(Gerd Pickard)

```
11 a$=left$(ti$,2):b$=mid$(ti$,3,2):c$=r
ight$(ti$,2):print"■";a$;":";b$;":";c$:g
oto11
```

Ein RENEW, das funktioniert (C 64/VC 20)

Routinen, die nach einem Reset oder NEW das Basic-Programm zurückholen, sind zwar schon oft veröffentlicht worden, doch waren die meisten entweder nicht lauffähig oder das Abtippen wurde durch die Länge zu einer umständlichen und unsicheren Prozedur. Dieser Einzeiler ist natürlich im Direktmodus, also ohne Zeilennummer, einzugeben. Vorher dürfen jedoch keine Variablen definiert oder Basic-Zeilen eingetippt werden, da sonst das gelöschte aber noch im Speicher befindliche Programm zerstört werden würde. Das Prinzip des UNNEW-Programms beruht auf dem Aufruf einer System-Routine, die die Basic-Zeilen neu bindet und das Ende des gelöschten Programms herausfindet. Der erste POKE dient nur dazu, der Routine vorzutäuschen, daß sich noch ein ungelöschtes Basic-Programm im Speicher befindet, da sie sonst nicht arbeitet. Die Endadresse des Basic-Programms wird um zwei erhöht und in den Zeiger auf den Start der Variablen (45/46) übertragen. Der CLR-Befehl gleicht alle weiteren Basic-Zeiger diesem Wert an.

SYS-Adresse für VC 20: 50483. Da beim VC 20 die Startadresse des Basic-Programms je nach Speicherausbau verschieden ist, muß man die Zahl »2050« durch das Ergebnis des folgenden Terms ersetzen« PRINT/PEEK(43)+256*PEEK(44)+1.

(Hartmut + Martin Sprave)

```
6 rem c-64
7 poke2050,8:sys42291:poke46,peek(35)-(p
eek(781)>253):poke45,peek(781)+2and255:c
lr
8 rem   vc-20:
9 rem   wie oben,aber sys50483  und
10 rem  statt 2050: poke(43)+256*poke(44
)+1

ready.
```

Eine Zeile — kompletter Datenschutz (C 64)

Dieses Programm verhindert das Auflisten des Inhaltsverzeichnisses jeder beliebigen Diskette. Nebenbei stellen sich noch einige sehr brauchbare Effekte ein. Das Löschen und Überschreiben der auf der Diskette befindlichen Programme ist nicht mehr möglich. Außerdem kann nichts mehr auf die Disk geschrieben werden. Ebenso werden Diskettenbefehle wie VALIDATE oder INITIALIZE ignoriert. Sogar ein Headern der Diskette ohne neue ID ist nicht möglich. Das einzige, was den Inhalt der Diskette noch manipulieren kann, ist das Headern (Diskettenbefehl »NEW«) mit einer neuen ID-Nummer.

Das Laden der bereits auf der Disk befindlichen Programme funktioniert hingegen ganz normal. Allerdings sollte man sich die Namen der gespeicherten Files merken, denn das

Commodore 64 und Software fürs Büro

Der C 64 wird 100 000fach eingesetzt zum Spielen, Lernen und auch für Büroanwendungen. Mit ausgereifter Software wird der C 64 zum idealen Organisationsmittel im Büro — für kleine bis mittlere Ingenieurbüros, Einzel- und Großhändler, gewerbliche Betriebe und Fachabteilungen in Großunternehmen. Überall da, wo viele Briefe geschrieben, Daten und Adressen verwaltet und Kalkulationen gerechnet werden.



Mit diesen 4 Programmen können Sie bis zu 80% Ihrer Büroanwendungen mit dem C 64 realisieren — auch als Computer-Laie!



M&T-Kartei

- bis zu 40 Einträge pro »Karteikarte« (Datensatz)
- Anzahl der frei definierten Grundmasken und »Karteikarten« hängt lediglich von der Diskettenkapazität ab
- integrierte Rechenmöglichkeiten durch Hilfsspeicher
- Datenzugriff über jeden Schlüssel möglich oder sequentiell
- Daten sind ständig nach 3 Kriterien vorsortiert (hohe Zugriffsgeschwindigkeit)
- Listenausgabe wahlweise auf Bildschirm oder Drucker
- wählbare Zeilenbreite zwischen 40 und 100 Zeichen pro Listenzeile (deutscher Zeichensatz)
- komfortable Dienstprogramme (Formatieren, Inhaltsverzeichnis, Korrektur)
- hohe Datensicherheit durch programmgesteuerte Datensicherung
- automatisches Zurückschreiben von geänderten Datensätzen in die jeweilige Datei

Best.-Nr.MD 218A DM 179,—* (Sfr. 165,—)

M&T-Kalkulation

- einfache Bedienung durch professionelle Fenstertechnik und Funktionstastensteuerung
- ausreichende Tabellenkapazität mit 1375 belegbaren Feldern
- gesonderte Darstellung verschiedener Tabellenausschnitte
- Zahleneingabe bis zu neun Nachkommastellen
- vielfältige Berechnungsarten und -schemen
- wahlweise Rechenautomatik oder Berechnung per Tastendruck
- frei definierbare Tabellenbereiche für den listenmäßigen Ausdruck
- englischer und deutscher Zeichensatz
- programmgesteuerte Datensicherung
- Schnittstelle zu M&T-Text

Best.-Nr.MD 217A DM120,—* (Sfr. 110,50)

*inkl. MwSt.

Unverbindliche Preisempfehlung

M&T-Textverarbeitung

- übersichtliche Menü- und Funktionstastensteuerung
- horizontales und vertikales Bildschirm-Scrolling
- umfangreiche Editiermöglichkeiten (automatische Trennvorschläge)
- Tabulatorfunktionen
- variable Zeilenbreite bis 80 Zeichen
- individuelle Farbeinstellung
- Druckmodus kann in jeder Zeile geändert werden (linksbündig, rechtsbündig und Blocksatz)
- Schnittstelle zu M&T-Adreß

Best.-Nr.MD 180A DM129,—* (Sfr. 119,—)

M&T-Adressverwaltung

- einfache Bedienung durch Bildschirmmasken
- Speicherkapazität von 622 Adressen
- Ausdruck nach beliebigen Suchbegriffen
- Ausdruck auf Endlospapier oder Adreßaufkleber
- bedienerfreundliche Adreßpflege

Optimale Ergänzung zu M&T-Text
- Erstellung von Serienbriefen
- individuelle Empfängeradresse
- persönliche Briefanrede

Best.-Nr.MD 181A DM79,—* (Sfr. 73,—)

Alle Programme von M&T laufen auf denselben minimalen Hardwareanforderungen:

- Commodore 64 • Diskettenlaufwerk • beliebiger Commodore- oder ASCII-Drucker

M&T-Programme sind

- ohne Vorkenntnisse bedienbar
- in Maschinensprache geschrieben
- menügesteuert
- Diskettenversionen
- mit ausführlichen Bedienungsanleitungen und
- durch separate Datendisketten in sich erweiterbar.

Stellen Sie Ihren Commodore 64 als persönlichen Büromanager ein!

In guten Buchhandlungen, Computershops und Fachabteilungen der Kaufhäuser.
Sollten diese Programme dort nicht erhältlich sein, benutzen Sie bitte die Bestellkarte in diesem Heft.

Markt&Technik
Verlag Aktiengesellschaft
Buchverlag

Hans-Pinsel-Straße 2, 8013 Haar bei München, ☎ (089) 4613-220
Schweiz: Markt & Technik-Vertriebs AG, Alpenstraße 14, CH-6300 Zug, ☎ 042/223155
Österreich: Rudolf-Lechner & Sohn, Heizwerkstraße 10, A-1232 Wien, ☎ 0222/677528

Suchen mit Jokerzeichen ist bestenfalls ein netter Zeitvertreib, aber nicht unbedingt immer erfolgreich. Hat man beispielsweise zwei Programme, die mit A anfangen, auf der Diskette, so muß man von dem zweiten Programm ja mindestens die ersten beiden Buchstaben angeben, um es laden zu können.

Daß das Directory nicht mehr gelistet werden kann liegt daran, daß es als Basic-Programm geladen wird und in dem veränderten Directory drei Nullen am Anfang erscheinen. Dies ist für den Interpreter jedoch das Zeichen für das Programmende. Das »Directory-Programm« endet also bereits nach zehn Bytes. Die ersten fünf Bytes stellen den Zeilenanfang und die Zeilennummer dar. Darauf folgen ein Leerzeichen und ein Anführungszeichen, die ja immer am Anfang eines Inhaltsverzeichnisses stehen. Um diese drei Zeichen beim Auslisten verschwinden zu lassen, folgen nun drei chr$(20), die jeweils ein Delete darstellen. Dies bewirkt, daß das Directory beim Auslisten nun völlig verschwindet. Das Listen wird hier abgebrochen, da nun unmittelbar die drei Nullen folgen, die das Programmende markieren. So erfolgt auf den List-Befehl nur die Meldung »READY«.

Die reversen »t« werden am einfachsten eingegeben, wenn man zuerst zwei Anführungszeichen hintereinander schreibt, dann den Cursor zurück auf das zweite Anführungszeichen bewegt, dreimal die Taste »INST« (= Insert) und danach dreimal die Tasten »SHIFT« und »INST« (= Delete) betätigt. Die Zeile kann natürlich genauso im Direktmodus abgeschickt werden.

(Volker Ritzhaupt)

```
1 open1,8,3,"#":open2,8,15,"b-p3,144":pr
int#1,"ttt"chr$(0)chr$(0)chr$(0):print#2
,"u2:3,0,18":print#2,"i
2 rem
3 rem verhindert auflisten des directory
       und macht schreibschutz auf disk
4 rem
```

Geänderter Zeichensatz (C 64)

Das Programm enthält eine Variable, sechs POKE-Befehle, einen PEEK-Befehl und eine FOR-NEXT-Schleife. In der Variablen R wird die Adresse des Interrupt-Registers definiert (#56334/$DC0E). Nun wird die Position des Zeichengenerators geändert, indem Bit 3 der Adresse 53272 ($D018) gesetzt und Bit 1 gelöscht wird. So wird der Bereich ab #8192 beziehungsweise $2000 ausgewählt. Um den alten Zeichensatz lesen zu können, muß der Interrupt ausgeschaltet werden (Bit 0 in Adresse 56334 gelöscht), ebenso der Video-Chip (Bit 2 in Adresse 1 gelöscht). Die folgende Schleife würde in einem übersichtlichen Programm so aussehen

FOR I = 0 TO 4095
POKE 8192+I,
PEEK(I+53248) AND 60
NEXT

Es wird also der Zeichensatz von Adresse #53248 beziehungsweise $D000 an nach #8192 übertragen, wobei jedes Byte durch die AND-Funktion geändert wird. Diese bewirkt, daß die Bits 0, 1, 6 und 7 in jedem Byte gelöscht werden, also ganz einfach jedem Zeichen der rechte und der linke Rand abgeschnitten wird. In meinem Programm überdeckt die Schleife einen etwas größeren Bereich als eigentlich nötig wäre, nämlich von 6hoch5 = 7776 bis R/5 = 11267, was dem Resultat jedoch keinen Abbruch tut.

Schließlich werden Video-Chip und Interrupt wieder eingeschaltet (Bit 2 in Adresse 1 und Bit 1 in Adresse 56334 gesetzt).

Bemerkenswert ist vielleicht, daß ich nach der Realisierung meiner Idee ein einziges Zeichen zuviel im Programm hatte. Nach drei Stunden angestrengten Tüftelns kam mir die erlösende Idee. Ursprünglich hatte ich den Video-Chip durch POKE1,55 wieder eingeschaltet. Es genügt aber auch POKE1,7, da die übrigen Bits vom Prozessor automatisch gesetzt werden.

(Klaus Vorwalter)

```
1 r=56334:poke53272,24:poker,0:poke1,51:
fori=6↑5tor/5:pokei,peek(i+45056)and60:n
ext:poke1,7:poker,1
2 rem
```

Trick 17 mit ON..GOTO (C 64/VC 20)

Eine sehr interessante Version einer ON..GOTO-Anweisung. Zur Erklärung braucht eigentlich nur gesagt werden, daß der Ausdruck (A$="A") den Wert —1 hat, wenn ein »A« eingegeben wurde, sonst den Wert 0. Rechnen Sie nach oder probieren Sie es aus: Es funktioniert einwandfrei.

(Peter Zankl)

```
100 rem tastaturabfrage mit sprung
200 rem :
300 rem vorher:
400 :
410 geta$:ifa$=""then410
420 ifa$="a"then2000:rem programmteil a
430 ifa$="b"then3000:rem programmteil b
440 ifa$="x"then end:rem ende
450 goto 410
499 :
500 rem nachher:
600 :
610 geta$:on1-(a$="a")-2*(a$="b")-3*(a$=
"x")goto610,2000,3000:end
620 :
```

Das kürzeste Abenteuerspiel der Welt (C 64/VC 20)

Das Spiel heißt »Nosferatu«. Im Spiel selbst stehen Sie einem Vampir gegenüber und müssen herausfinden, wie Sie ihn besiegen. Erlaubt sind deutsche Zweiwortkommandos. Eine Alternative für die Lösung besteht, indem Sie anstatt »WIRF KNOBLAUCH« »ZEIGE KREUZ« eintippen. Sie können sich auch andere Situationen überlegen, was das Programm noch zum kürzesten Adventure-Generator macht Cursorsteuerzeichen: Reverses Q = Cursor hoch, Reverses q = Cursor runter. Bitte achten Sie auf mein Copyright (für einen Vermerk war leider kein Platz) DARUM HIER: (C) Copyright 1984, by me. Und noch etwas: Mogeln Sie beim Abtippen bitte nicht! Es wäre schade, wenn Sie schon beim Eintippen die Lösung kennen würden.

Vampirische Grüße
(Thomas Werner)

```
1 print"ein vampir!":inputa$:print"sieg!
":ifa$<>"wirf knoblauch"thenprint"[illegible]klapp
t nicht![illegible]":goto1
2 rem
```

Dividieren mit beliebig vielen Stellen hinter dem Komma (V 64)

Das Programm muß mit den bekannten Abkürzungen eingegeben werden, damit es in das 80-Zeichenformat paßt.

Variablenliste
Z = Zahl (mathematisch korrekt : DIVISOR)
D = Dividend
N = Anzahl der gewünschten Stellen hinter dem Komma
E% = Ergebnis ohne Rest (wird ausgegeben)
R = Rest
Q% = Quotient beschränkt auf Vorkommazahl

Beschreibung:

Um die Vorkommazahl zu erhalten, benötigt man Q = INT(Z/D), was ich durch Q% = Z/D vereinfacht habe. Q% wird als Vorkommazahl ausgegeben. Es wird der Rest, nämlich die Differenz aus Z und Q% * D gebildet. Der Rest dient als Ausgangspunkt für die folgende Berechnung. Die Schleife wird Nmal durchlaufen, um exakt N Stellen hinter dem Komma zu ermitteln.

Analog zum Vorschleifenteil wird E% = R * 10/D gebildet. E% wird ausgegeben als 1.te Stelle.

Der neue Rest wird durch $R_{(neu)} = 10 * R_{(alt)} - D * E\%$ (vergleiche Vorkommateil) gebildet.

Und wiederum kann die Schleife durchlaufen werden, bis alle Stellen ausgegeben sind.

Beispiel:

Wie lautet die 85. Stelle hinter dem Komma von $^{116}/_{13}$? (also: Z = 116, D = 13, N = 85) Programmlauf ....
Ergebnis:

Die 85. Stelle hinter dem Komma ist 9. (Selbstverständlich werden auch alle übrigen Stellen ausgegeben).

(Heinz Bauschke)

```
0 inputz,d,n:q%=z/d:printq%:r=z-d*q%:for
i=1ton:e%=r*10/d:print"[illegible]";e%;:r=10*r-d*e
%:next
10 rem
```

Simulation — GOTO X (C 64)

Das Programm simuliert den im Commodore-Basic nicht vorhandenen Befehl GOTO X.

Zu dem etwas seltsamen Aussehen der Zeile läßt sich folgendes sagen: In den Anführungszeichen steht ein Maschinenprogramm, dessen Opcodes im Listing des Basic-Interpreters diese merkwürdigen Zeichen erzeugen. Das disassemblierte Maschinenprogramm finden Sie in Listing 2, wobei die absoluten Adressen keine Bedeutung haben, das Programm adressiert nur relativ. Der mit der Materie etwas vertrautere Leser wird feststellen können, daß das Programm etwas umständlich geschrieben ist. Das liegt daran, daß nicht alle Character, die ein Opcode darstellen, eindeutig einer Zahl zugeordnet sind. So gäbe zum Beispiel die Zahl 255 im Maschinenprogramm das Zeichen (Commodore & K), wenn die Zeile aber dann editiert würde, interpretiert das Basic dieses Zeichen aber mit 161. So ließe sich die Zeile nicht editieren, wäre also keine Basic-Zeile. Aus diesem Grund wurden alle solchen »doppeldeutigen« Codes umgangen, was die Umständlichkeit des Maschinenprogramms zur Folge hatte. Die SYS-Anweisung stellt die Lage des Maschinenprogramms mit Hilfe zweier Zeiger des Interpreters fest und startet das Programm.

Nun zur Benutzung der GOTO X-Zeile und wie man sie eingibt: Es ist zwar möglich, die Zeile mit der in Ausgabe 7/84 beschriebenen »Finkel«-Methode einzugeben, was jedoch umständlich, zeitaufwendig und fehleranfällig wäre. Deshalb ist in Listing 3 ein Generierungsprogramm abgebildet. Dieses wird eingegeben und gestartet.

Wenn der Checksum-Test positiv ausgefallen ist und man sich vergewissert hat, daß das Programm genauso eingegeben wurde, wie es ausgelistet ist, gibt man LIST ein, gefolgt von NEW. Dann fährt man mit dem Cursor auf Zeile 10 und drückt RETURN. GOTO X steht nun im Speicher und kann gelistet, editiert und abgeSAVEt werden.

Selbstverständlich kann die Zeilennummer zwecks der Einbindung in eigene Programme geändert werden. Dazu weist man der Variablen LL% den Wert der anzuspringenden Zeile zu und springt mit GOTO zur Zeile, in der GOTO X steht.

(Reinhard Jurk)

```
10 syspeek(61)+256*peek(62)+26:"[illegible]"
20 rem
30 rem goto x: 11%=zeilennummer:goto10
40 rem
```

Listing 1

```
C*
     PC  IRQ  NV-BDIZC AC XR YR SP
.;  E147 EA31 00110001 08 08 02 F6
.
.,  081A  A9 CC       LDA #$CC
.,  081C  85 45       STA $45
.,  081E  85 46       STA $46
.,  0820  A5 9D       LDA $9D
.,  0822  85 55       STA $55
.,  0824  85 56       STA $56
.,  0826  A0 0C       LDY #$0C
.,  0828  C8          INY
.,  0829  91 55       STA ($55),Y
.,  082B  A2 81       LDX #$81
.,  082D  CA          DEX
.,  082E  C8          INY
.,  082F  8A          TXA
.,  0830  91 55       STA ($55),Y
.,  0832  AC 80 A3    LDY $A380
.,  0835  A6 2E       LDX $2E
.,  0837  A5 2D       LDA $2D
.,  0839  20 11 B1    JSR $B111
.,  083C  20 2B AF    JSR $AF2B
.,  083F  20 01 B8    JSR $B801
.,  0842  4C A3 A8    JMP $A8A3
.
.
READY.
```

Listing 2

```
             G O T O  X

  BITTE UNBEDINGT GENAUSO EINGEBEN !!!
  43 KLAMMERAFFEN (@)


5 GOTO20
10 SYSPEEK(61)+256*PEEK(62)+26:"@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@"
20 FORA=2082TO2124:READX:S=S+X:POKEA,X:N
EXT:IFS<>5185THENPRINT"DATA ERROR !":END
30 PRINT"OK."
40 DATA169,204,133,69,133,70,165,157,133
,85,133,86,160,12,200,145,85,162,129
50 DATA202,200,138,145,85,172,128,163,16
6,46,165,45,32,17,177,32,43,175,32,1
60 DATA184,76,163,168

READY.
```

Listing 3. Der Basic-Lader für GOTO X

Irrgarten (VC 20/C 64)

Das Programm belegt nur 40 Bytes Speicherplatz. Es erzeugt einen zufallsbedingten Irrgarten und läuft auf einem C 64 oder VC 20:

Zum Programm: X ist eine durch die RND-Funktion ermittelte Zufallszahl. Sie ist nicht gerundet und liegt zwischen 0 und 2. Der Computer druckt hierauf den CHR$-Code von 164 plus der Zufallszahl x aus, wobei er von sich aus X auf 0 oder 1 rundet. Es erscheint also auf dem Bildschirm je nach X das CHR$-Zeichen von 164 oder 165.

Der nach dem PRINT-Befehl folgende Strichpunkt bewirkt, daß das nächste Zeichen nicht erst in der nächsten Reihe, sondern gleich neben dem Zeichen davor erscheint.

Die Variable Q, welche durch den RUN-Befehl automatisch gleich Null gesetzt wurde, wird nun um 1 erhöht. Ist sie noch kleiner als 880, so wird die Schleife erneut durchlaufen. Das entstehende horizontale und vertikale Muster ist ein Labyrinth — aufgebaut aus nur diesen beiden Zeichen.

Durch Verwendung anderer CHR$-Zeichen (zum Beispiel 176 bis 179) läßt sich das Programm gut variieren.

Jedenfalls: Es ist nicht immer einfach den richtigen Weg zu finden!

(Marco Gleiter)

```
1 x=rnd(1)*2:printchr$(164+x);:q=q+1:ifq
<880thengoto1
2 rem
```

Clevere Idee: Grafikbildschirm löschen mit dem DIM-Befehl (C 64)

Programmbeschreibung:

1. a=0:b=0

Im weiteren Verlauf wird das Variablenfeld beeinflußt. Die Variablen a und b müssen deshalb vor der DIM-Anweisung angelegt sein.

2. a=peek(49):b=peek(50)

Die Werte für das Variablenende werden gesichert.

3. dim f((16191-a-b*256) /5)

Durch das Dimensionieren einer Variablen wird ein entsprechend großer Platz hinter dem Variablenende freigemacht beziehungsweise mit dem Wert 0 gefüllt. Nun muß der Platz nur noch mit der Adresse des Bildschirms übereinstimmen.

Zur Formel (16191-a-b*256) / 5 :

Der Bildschirmspeicher liegt von 8 192 bis 16 191. Die Variable f wird hinter dem bisherigen Variablenende angelegt.
Adresse des Variablenendes: a+b*256
also: 16191-a-b*256
Pro indizierte Variable werden 5 Byte freigemacht,
also: (16191-a-b*256) / 5

Die mit 0 indizierte Variable ist nicht berücksichtigt, ebenso die ersten 7 Byte für Variablenname und Dimension.

4. poke 49,a : poke 50,b

Die alten Werte werden wieder hergestellt. Die Variable f ist jetzt nicht mehr dimensioniert.

Zum ordnungsgemäßen Verlauf müssen zwei Punkte beachtet werden.

1. Das Variablenende muß kleiner als 8186 sein, das heißt das Programm darf einschließlich Variablenfeld nicht größer als 6 KByte sein. Mit 6 KByte steht jedoch ein ausreichend großer Platz zur Verfügung.

Ist dies nicht der Fall, wird lediglich der Bildschirm nicht ganz gelöscht. Das Programm kann in keinem Fall Schaden nehmen.

2. Soll das Grafikprogramm zum Abspeichern von Bildern benutzt werden, darf man nicht vergessen, den Zeiger für Speichergrenze (peek(55)+peek(56)*256) auf eine Adresse kurz hinter den Bildschirmspeicher zu setzen.

In meinem Grafikprogramm rufe ich die Zeile 10 immer dann auf, wenn über die Tastatur »shift + clr home« eingegeben wird (entsprechend »freier Bildschirm im Textmodus«).

(Manfred Hedtke)

```
10 a=0:b=0:a=peek(49):b=peek(50):dimf((1
6191-a-b*256)/5):poke49,a:poke50,b
20 rem
```

Zweifarbiger Rahmen (C 64)

Wie jeder weiß, ist der Bildaußenrahmen des C 64 immer einfarbig. Ich habe nun einen kleinen Einzeiler geschrieben, der einen zweifarbigen Außenrahmen simuliert. Die beiden Farben sind: 0 = schwarz und 1 = weiß. Es ist darauf zu achten, daß die Basic-Zeile genauso eingetippt wird, wie sie abgebildet ist. Ich meine damit die Anzahl der Spaces und Doppelpunkte. Erklärung: Ich wollte die Grenze zwischen den beiden Farben möglichst ruhig auf dem Bildschirm haben und mußte dazu die genaue Farbwechselperiode finden. Der Interpreter arbeitet Doppelpunkte und Spaces unterschiedlich schnell ab; so konnte ich mit den Spaces eine Feineinstellung vornehmen. Aber vorsichtig: Sobald während des Programmlaufs eine Taste gedrückt wird, tritt die Raster-Interrupt Tastaturabfrage in Kraft und das Bild fängt an zu »laufen«.

Hilfe zum Abtippen:
Zwischen den beiden POKEs, und den Doppelpunkten befinden sich je fünf Spaces. Es folgen dann 17 beziehungsweise 15 Doppelpunkte.

So einfach es auch immer ist: Es verblüfft einen doch!

(Markus Hillebrand)

```
10 poke53280,1     :::::::::::::::::poke
53280,0     :::::::::::::::goto10
20 rem
```

Fakultät (C 64)

Das Programm berechnet Fakultäten, ist also sehr nützlich für einige mathematische Funktionen, da der C 64 keinen derartigen Befehl besitzt.
Variablen:
A = Eingabevariable,
B = Zählvariable,
C = Rechen- und Ausgabevariable.

Das Programm berechnet sogar Fakultäten, die größer als »69« sind. Es ist also besser als ein Taschenrechner.

(Detlef Marks)

```
10 rem fakultaeten
20 :
30 inputa:frb=1toa:c=c+log(b):next:c=c/l
og(10):print10↑(c-int(c));"e";int(c):run
40 :
```

Fortsetzung von Seite 141

Bildwiederholspeichers bei der Erzeugung eines Videosignals. Die ausgelesenen Werte werden durch einen Digital-Analog-Wandler in eine Folge von Spannungswerten verwandelt, die nach Durchlaufen eines geeigneten Tiefpasses ein akustisches Signal ergeben. Mit dieser Technik ist prinzipiell jedes akustische Ereignis erfaßbar und reproduzierbar.

Rein digitale **Signalverarbeitung** ist das allgemeinste und universellste Syntheseprinzip, das vorstellbar ist. Wie beim Sound-Sampling wird das Signal als Folge von Digitalwerten repräsentiert. Diese Folge wird aber nicht unbedingt nur durch Auslesen eines Speichers gewonnen, sondern kann auch in Realzeit errechnet werden. Die Tragweite dieses Konzepts besteht darin, daß prinzipiell jedes andere Syntheseverfahren durch Signalverarbeitung nachgebildet werden kann. Synthesen, die mit anderen Mittel kaum zu realisieren sind, wie die Erzeugung künstlicher Sprache, werden fast ausschließlich mit digitaler Signalverarbeitung realisiert. Die Crux an der Signalverarbeitung ist der enorm hohe Bedarf an Rechengeschwindigkeit und Genauigkeit, der von Universal-Mikroprozessoren noch nicht befriedigt werden kann. Für ein hochwertiges Signal im Audiobereich benötigt man mindestens 40000 Abtastwerte pro Sekunde, die außerdem noch genauer als 8 Bit sein sollten. Für die Berechnung eines Abtastwertes bleiben damit 25 Mikrosekunden, eine Zeit, in der zum Beispiel die CPU 6502 (6510) zirka 5 bis 10 Befehle abarbeitet, viel zu wenig für die Erzeugung selbst einfachster Kurvenformen. Es gibt seit einiger Zeit bereits hochintegrierte Signalprozessoren. Das sind spezialisierte CPUs oder sogar Einchip-Prozessoren, die mit einem meist einfachen Befehlssatz ausgestattet sind und hohe Verarbeitungsgeschwindigkeiten erreichen (Befehlsausführungszeit 100—200 ns im Gegensatz zu mehreren µs bei Universal-Mikroprozessoren). Diese Bausteine sind aber teuer, schwer erhältlich, schwierig zu handhaben und nicht so universell einsetzbar wie Standard-Prozessoren. Den Sound-Chip im C 64 kann man bereits als einen kleinen Signalprozessor ansehen, dessen Betriebsprogramm allerdings fest vorgegeben ist. Man kann seinen Ablauf nur über die bereits erwähnten Steuerregister beeinflussen.

Ziel dieser Serie ist es zu zeigen, wie man mit geeigneter Steuersoftware möglichst viel aus diesem Baustein herausholt.

(Thomas Krätzig/aa)

Inserentenverzeichnis

Impressum

Herausgeber: Carl-Franz von Quadt, Otmar Weber

Chefredakteur: Michael M. Pauly (py)
Stellv. Chefredakteur: Michael Scharfenberger (sc)
Redakteure: aa = Albert Absmeier, leitender Redakteur, ev = Volker Everts, gk = Georg Klinge, rg = Christian Rogge
Redaktionsassistenz: Gerda Siegl (202)
Fotografie: Janos Feitser, Titelfoto: Alex Kempkens

Layout: Leo Eder (Ltg.), Dagmar Berninger, Willi Gründl, Cornelia Weber

Auslandsrepräsentation:
Schweiz: Markt&Technik Vertriebs AG, Alpenstrasse 14, CH-6300 Zug, Tel. 042-2231 55/56, Telex: 862329 mut ch
USA: M & T Publishing, 2464 Embarcadero Way, Palo Alto, CA 94303; Tel. (415) 424-0600; Telex 752351

Manuskripteinsendungen: Manuskripte und Programmlistings werden gerne von der Redaktion angenommen. Sie müssen frei sein von Rechten Dritter. Sollten sie auch an anderer Stelle zur Veröffentlichung oder gewerblichen Nutzung angeboten werden, so muß dies angegeben werden. Mit der Einsendung von Manuskripten und Listings gibt der Verfasser die Zustimmung zum Abdruck in von der Markt & Technik Verlags AG herausgegebenen Publikationen und zur Vervielfältigung der Programmlistings auf Datenträger. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte und Listings wird keine Haftung übernommen.

Herstellung: Klaus Buck (180)

Anzeigenverkauf: Brigitta Fiebig (211)

Anzeigenverwaltung und Disposition: Michaela Hörl (171)

Anzeigenformate: 1/1-Seite ist 266 Millimeter hoch und 185 Millimeter breit (3 Spalten à 58 mm oder 4 Spalten à 43 Millimeter). Vollformat 297 x 210 Millimeter. Beilagen und Beihefter siehe Anzeigenpreisliste.

Anzeigenpreise: Es gilt die Anzeigenpreisliste Nr. 2 vom 1. Januar 1985.
Anzeigengrundpreise: 1/1 Seite sw: DM 8500,-. Farbzuschlag: erste und zweite Zusatzfarbe aus Europaskala je DM 1400,-. Vierfarbzuschlag DM 3800,-. Plazierung innerhalb der redaktionellen Beiträge: Mindestgröße 1/2-Seite
Anzeigen im Computer-Markt: Die ermäßigten Preise im Computer-Markt gelten nur innerhalb des geschlossenen Anzeigenteils, der ohne redaktionelle Beiträge ist. 1/1-Seite sw: DM 6400,-. Farbzuschlag: erste und zweite Zusatzfarbe aus Europaskala je DM 1000,-. Vierfarbzuschlag DM 3000,-. **Anzeigen in der Fundgrube:** **Private Kleinanzeigen** mit maximal 5 Zeilen Text DM 5,- je Anzeige.
Gewerbliche Kleinanzeigen: DM 10,- je Zeile Text.
Auf alle Anzeigenpreise wird die gesetzliche MwSt. jeweils zugerechnet.

Vertriebsleitung, Werbung: Hans Hörl (114)

Vertrieb Handelsauflage: Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz: Pegasus Buch- und Zeitschriften-Vertriebsgesellschaft mbH, Hauptstätterstraße 96, 7000 Stuttgart 1, Telefon (07 11) 6483-0

Erscheinungsweise: 64'er, Magazin für Computerfans, erscheint monatlich, Mitte des Vormonats.

Bezugsmöglichkeiten: Leser-Service: Telefon 089/46 13-1 19. Bestellungen nimmt der Verlag oder jede Buchhandlung entgegen. Das Abonnement verlängert sich zu den dann jeweils gültigen Bedingungen um ein Jahr, wenn es nicht zwei Monate vor Ablauf schriftlich gekündigt wird.

Bezugspreise: Das Einzelheft kostet DM 6,50. Der Abonnementspreis beträgt im Inland DM 78,- pro Jahr für 12 Ausgaben. Darin enthalten sind die gesetzliche Mehrwertsteuer und die Zustellgebühren. Der Abonnementspreis erhöht sich um DM 18,- für die Zustellung im Ausland, für die Luftpostzustellung in Ländergruppe 1 (z.B. USA) um DM 38,-, in Ländergruppe 2 (z.B. Hongkong) um DM 58,-, in Ländergruppe 3 (z.B. Australien) um DM 68,-.

Druck: E. Schwend GmbH, Schmollerstr. 31, 7170 Schwäbisch Hall

Verantwortlich: Für redaktionellen Teil: Michael M. Pauly.
Für Anzeigen: Hannelore Schmidt.

Redaktions-Direktor: Michael Pauly

Vorstand: Carl-Franz von Quadt, Otmar Weber

Anschrift für Verlag, Redaktion, Vertrieb, Anzeigenverwaltung und alle Verantwortlichen:
Markt&Technik Verlag Aktiengesellschaft, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon 089/46 13-0, Telex 522052

Mitteilung gem. Bayerischem Pressegesetz: Die Rechtsform wurde von Gesellschaft mit beschränkter Haftung in Aktiengesellschaft geändert. Aktionäre, die mehr als 25% des Kapitals halten: Otmar Weber, Ingenieur, München; Carl-Franz von Quadt, Betriebswirt, München, Aufsichtsrat: Dr. Robert Dissmann (Vorsitzender), Karl-Heinz Fanselow, Eduard Heilmayr.

Telefon-Durchwahl im Verlag:

Wählen Sie direkt: Per Durchwahl erreichen Sie alle Abteilungen direkt. Sie wählen 089-46 13 und dann die Nummer, die in Klammern hinter dem jeweiligen Namen angegeben ist.



Die nächste Ausgabe erscheint am 18. Januar

Assembler Vergleichstest (2)

Im zweiten Teil unseres Vergleichstests nehmen wir die Assembler bis 100 Mark unter die Lupe. Sie besitzen vielleicht nicht die Leistungsfähigkeit der »großen«, aber reicht nicht auch ein preiswertes Programm?

Grafikkurs — noch mehr Informationen

Es hat sich gezeigt, daß viele Leser noch mehr zur Grafikprogrammierung wissen wollen. Deshalb wird der Grafik-Kurs in lockerer Reihenfolge fortgesetzt. Wir gehen auf spezielle Wünsche der Leser ein und stellen Ihnen ein Grafik-Hilfsprogramm vor, das viel mehr ist, als nur eine einfache Hilfe. Um leichter Grafik programmieren zu können, stehen auch nützliche Befehle wie OLD, MERGE, RENUMBER, AUTONUMBER, etc. zur Verfügung.

Basic-Compiler im Test

Basic-Compiler stellen ein Bindeglied zwischen reinen Basic-Programmen und Maschinensprache her. Sie beschleunigen den Ablauf der Basic-Programme, erreichen jedoch nie die Schnelligkeit von Assembler. Compiler bringen aber nicht nur eitel Freude, sondern einen auch manchmal zum Verzweifeln. Anders gesagt, sie haben ihre Macken, die man kennen muß. Wir stellen Ihnen die wichtigsten Compiler vor.

Außerdem...

— Sieben lehrreiche Kurse für Anfänger und Profis
— Tests verschiedener Sprachausgabesysteme
— interessante Listings zum Abtippen
— und wieder viele Tips und Tricks für VC 20 und C 64

Compiler

Im Mikrocomputerbereich setzen sich Compiler für die verschiedensten Sprachen immer mehr durch. Für die meisten Anwender bleibt der Compiler aber ein undurchschaubares Ding, das auf geheimnisvolle Weise in der Lage ist, höhere Programmiersprachen in Maschinencode zu übersetzen. Wir zeigen, wie sie funktionieren, wie sie konstruiert werden und wo ihre Stärken und Schwächen gegenüber einem Interpreter liegen.

MSE — eine enorme Erleichterung

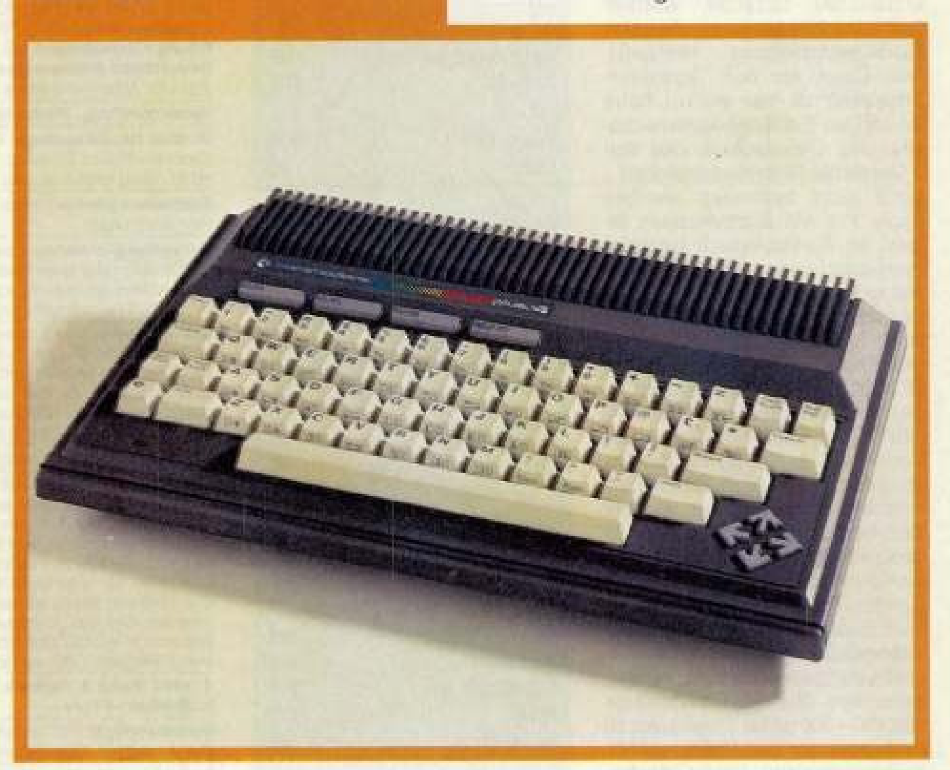
Ähnlich wie der »Checksummer« ist der MSE ein weiteres Hilfsmittel bei der Eingabe von Programmlistings, diesmal jedoch von reinen Maschinensprache-Programmen, das heißt beim Abtippen von DATAs. Der MSE verringert die Tipparbeit um ein Drittel und schließt Fehleingaben vollkommen aus. Durch akustische Meldungen können Sie DATAs sogar »blind« eingeben.

Plus/4 im Test

Seit kurzem kann man den Plus/4 in Computerläden entdecken. Was dieser neue Computer kann und für welchen Anwenderkreis er geeignet ist, zeigt unser Test.

Endlich: Die Lösung von Hobbit

Lange wurde auf die Lösung des Abenteuerspiels »Hobbit« gewartet. Einen der möglichen Lösungswege und den Lageplan finden Sie in der nächsten Ausgabe.



Mathematik-Lernprogramme

Zwölf verschiedene Mathematik-Lernprogramme wurden von uns getestet. Auch bei diesen Programmen war festzustellen: Nicht alles was als Lernsoftware verkauft wird, hat diesen Namen verdient. Welches der Programme eignet sich als »Nachhilfelehrer«?

Richtig verbunden

Wenn Sie einen Fernseher mit Videoeingang besitzen und den C 64 noch immer an der Antenne »hängen« haben, dann sollten Sie schon mal Ihren Lötkolben anheizen. Wir zeigen in einer Bauanleitung, wie der C 64, mit geringem Aufwand, an Fernsehgeräte und Monitore mit Videoeingängen angeschlossen werden kann.



Die unverzichtbare Verbesserung des C64: Spezial Rom
Der wesentlich verbesserte Umgang mit Ihrem C64.
Auf die Vorteile werden Sie bald nicht mehr verzichten wollen:

* Floppy wird doppelt so schnell
* automatisches Lesen des Fehlerkanals
* unterstützt Fast-Turbo (6x schnelleres Laden von Floppy)
* automatisches Listen der Directory
* Betriebssystem liegt sofort nach Einschalten des C64 vor, ohne daß vorher ein Programm geladen werden muß
* voll kompatibel

Nur 99.– DM



Floppy - Box für den C64
Endlich ist er da:
Der Schutz für Ihre C64-Anlage:

* stoßgeschützt
* stets transportbereit
* überall und ständig bedienbar
* leichter Ein- und Ausbau

Farbe: grau
technische Änderungen vorbehalten
Preis: 150.– DM
und zusätzlich die Discy - Box zur idealen Verfügbarkeit Ihrer Disketten: **10.– DM**



Das Modembuch
... zum Umgang mit Datenbanken
Mit Begriffserläuterungen, deutschsprachigem Datenbankführer, Mailboxen in Deutschland und im Ausland mit Telefonnummern; Normen, Gebühren, spezielle Anwendungen ...

... das Nachschlagewerk für Anfänger und Profis zur Datenfernübertragung mit Akustikkopplern und Modems – verständlich – praxisnah – umfangreich
Din A4 - Format 29.80 DM



Weitere Spitzenleistungen aus unserem Programm:
V 24 Anschlußkabel **nur 39.90 DM**
V 24 - Box für C64 auf Userport **nur 49.90 DM**

Händleranfragen erwünscht!
Bestellcoupon

Senden Sie mir die angegeben Artikel zu:

____	Stck. „Das Modembuch"	je 29.90 DM
____	Stck. „Spezial - Rom"	je 99.00 DM
____	Stck. V 24 Anschlußkabel	je 39.90 DM
____	Stck. V 24 - Box	je 49.90 DM
____	Stck. Floppy - Box	je 150.00 DM
____	Stck. Discy - Box	je 10.00 DM

Bei Vorauszahlung entfällt Nachnahmegebühr. Betrag + 2.50 VS; Ausland + 6.– DM VS.

(Unterschrift des Bestellers)

______________________ Name
______________________ Straße, Nr.
______________________ PLZ, Ort

An **Scientific Market, Manfred Hurth, Kottbusser Damm 29/30, 1 Berlin 61**
Tel.: 030 / 69 22 455


WARUM DAS HOBBY EINEN COMMODORE COMPUTER BRAUCHT.
Weil er dem Hobby-Sammler z. B. die ganze Kollektion katalogisiert.
Weil für den Musik-Fan ein leistungsstarker Synthesizer drin ist: 3 Stimmen mit je 8 Oktaven für heiße oder klassische Concertos con Commodore.
Weil er dem Hobby-Programmierer volle 38 KB RAM für BASIC-Programme und Daten bereitstellt. Oder satte 52 KB für Maschinensprache-Profis.
Weil der Commodore Heimcomputer natürlich auch die hochauflösende Farb-Grafik beherrscht. Samt Sprites für die Spielemacher.
Und weil der Spitzenreiter unter den Heimcomputern ein tüchtiger Mitstreiter bei jeder Art von Papierkrieg ist.
Darum braucht vielleicht nicht nur das Hobby einen Commodore Computer.
Beim Commodore-Vertragshandel, in führenden Warenhäusern, guten Rundfunk-, Fernseh- und Fotofachgeschäften und großen Versandhäusern.
Mehr Information und die Anschrift Ihres nächstgelegenen Commodore-Fachhändlers von: Commodore Büromaschinen GmbH, Abt. MK, Lyoner Str. 38, 6000 Frankfurt/M. 71. Oder per Telefon: Düsseldorf (02 11) 31 20 47/48 Frankfurt (0 69) 6 63 81 99 · Hamburg (0 40) 21 13 86 · München (0 89) 46 30 09 · Stuttgart (07 11) 24 73 29 Basel (0 61) 23 78 00 · Wien (02 22) 67 56 00.
Unsere BTX-Leitseite * 18919 #.
Commodore
Eine gute Idee nach der anderen.